

**Practical Hardware Details
for 8080, 8085, Z80, and 6800
Microprocessor Systems**

James W. Coffron

**Prentice-Hall, Inc.,
Englewood Cliffs
1981**

ДЖ. КОФФРОН

ТЕХНИЧЕСКИЕ
СРЕДСТВА
МИКРОПРОЦЕССОРНЫХ
СИСТЕМ

ПРАКТИЧЕСКИЙ
КУРС

Перевод с английского
канд. техн. наук В. А. Балыбердина,
канд. техн. наук Л. В. Шабанова

МОСКВА «МИР» 1983

ББК 32.973.2

К74

УДК 681.3

Коффрон Дж.

К74 Технические средства микропроцессорных систем: Практический курс. Пер. с англ.— М.: Мир, 1983.— 344 с., ил.

Книга американского специалиста представляет собой руководство по проектированию микропроцессорных систем на базе стандартных наборов распространенных семейств микропроцессоров 8080, 8085, Z80 и 6800. Детально анализируются особенности архитектуры и функционирования микропроцессоров и систем на их основе, обсуждаются преимущества и недостатки конкретных технических решений. Главное внимание уделено компоновке и испытаниям аппаратных средств. Изложение иллюстрируется значительным числом примеров построения реальных микропроцессорных систем.

Для специалистов, связанных с проектированием систем обработки данных на базе микропроцессоров.

К 2405000000-371
041(01)-83 158-83, ч. 1

ББК 32.973.2

Редакция литературы по новой технике

© 1981 by Prentice-Hall, Inc.,
Englewood Cliffs

© Перевод на русский язык, «Мир», 1983

Дж. Коффрон

ТЕХНИЧЕСКИЕ СРЕДСТВА МИКРОПРОЦЕССОРНЫХ СИСТЕМ

Научный редактор Т. Н. Шестакова. Младший научный редактор Е. П. Орлова. Художник Г. М. Чеховский. Художественный редактор Л. Е. Безрученков. Технический редактор Г. Б. Аюллина. Корректор Н. А. Гири

ИБ № 3741

Сдано в набор 03.01.83. Подписано к печати 13.05.83. Формат 60×90^{1/16}. Бумага типографская № 2. Объем 10,75 бум. л. Гарнитура литературная. Печать высокая. Усл. печ. л. 21,5. Усл. ир. отт. 21,5. Уч.-изд. л. 21,43. Изд. № 20/2110. Тираж 50 000 (1 завод 1—25 000) экз. Заказ 814. Цена 1 р. 30 к.

ИЗДАТЕЛЬСТВО «МИР».

129820, Москва, И-110, ГСП, 1-й Рижский пер., 2.

Московская типография № 11 Союзполиграфпром при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли.
Москва, 113105, Нагатинская ул., д. 1.

ПРЕДИСЛОВИЕ К РУССКОМУ ИЗДАНИЮ

Развитие и распространение микропроцессоров, а также всевозможных устройств и систем на их основе стимулируют постоянно возрастающий интерес к микропроцессорной тематике со стороны все более широкого круга технических специалистов. В последние годы в различных издательствах нашей страны был выпущен ряд монографий советских и зарубежных авторов, посвященных данной тематике. В этих работах нашел отражение круг вопросов, связанных с архитектурой микропроцессоров, структурой систем на их основе, возможностями и перспективами использования микропроцессорных устройств и систем в различных сферах жизни и деятельности человека и т. п. Можно полагать, что хотя спрос на литературу подобного рода удовлетворен еще не в полной мере, знакомство читателей с основами построения, функционирования и использования микропроцессорных систем уже состоялось. В этой связи становится понятной потребность в публикациях, которые давали бы ответы на многие специальные вопросы, возникающие у практических работников в процессе проектирования и отладки конкретной системы. Именно такую направленность имеет настоящая книга.

Главная особенность книги заключается в том, что основной акцент в ней смещен в сторону изложения практических вопросов, с которыми сталкиваются разработчики непосредственно в процессе создания системы. Книга в известной степени может рассматриваться как практическое руководство по проектированию и отладке микропроцессорных систем.

Другая особенность состоит в том, что при обсуждении вопросов технической реализации тех или иных устройств автор, как правило, не ограничивается рассмотрением какого-то одного, пусть предпочтительного инженерного решения, но анализирует различные подходы к решению, показывая их сильные и слабые стороны. Этому способствует проводимый автором анализ возможностей при использовании каждого из рассматриваемых типов микропроцессоров для реализации одних и тех же функций.

Особо следует остановиться на предлагаемой методике отладки технических средств системы, именуемой тестированием посредством статических сигналов. Она отличается простотой и наглядностью анализа коммуникационных процессов. Такая методика позволяет разработчику системы в полной мере осмыслить и прочувствовать все особенности взаимодействия микропроцессора с устройствами памяти и ввода-вывода.

Можно надеяться, что книга окажется весьма полезной для практических работников, связанных с проектированием и использованием микропроцессорных устройств и систем самого различного назначения.

В. А. Бальбердин

ПРЕДИСЛОВИЕ

Если вам необходима подробная информация практического характера по микропроцессорным системам, то ее можно найти в новой книге Дж. Коффрона.

Изложение материала начинается с описания обычной архитектуры системы с тремя шинами для 8-разрядных микропроцессоров. Это позволяет читателю хорошо усвоить организацию микропроцессорной системы, особенности ее функционирования и причины, по которым были выбраны те или иные технические решения. Книга написана простым языком, для нее характерна методически последовательная компоновка материала, что способствует глубокому изучению каждой из затронутых тем и позволяет использовать ее в качестве учебного пособия.

Книга является уникальной в том смысле, что в ней содержится описание полного набора схем для четырех широко распространенных микропроцессорных семейств — 8080, 8085, Z80, 6800,— поясняется назначение каждого вывода в корпусах интегральных схем, приводятся спецификации, соответствующие временные диаграммы, даются необходимые ссылки. Большое место отводится вопросам последовательной отладки технических средств системы, что обеспечивает их эффективное использование. Предлагаемый материал поможет читателю определять возможные неисправности в существующих микропроцессорах. Эта книга—продолжение ранее изданной монографии Дж. Коффрона «Устройство и отладка микропроцессоров», которая является своеобразным введением в микропроцессорные системы, предназначенным для специалистов по цифровой технике. Автору удалось заострить внимание на технических средствах микропроцессорных систем, что делает книгу весьма полезной в практическом отношении.

Многие читатели, вероятно, захотят приобрести эту книгу, чтобы использовать ее в качестве справочного пособия по техническим средствам ведущих микропроцессорных семейств.

В. Лонг

Профессор электроники
Футхилл колледж Лос-Алтос Хил,
Калифорния

ПРЕДИСЛОВИЕ АВТОРА

После завершения первого цикла занятий по микропроцессорам у студента обычно остается много неясного. Причина этого кроется в большом объеме изучаемого материала, освоение которого требует времени. Если по окончании такого цикла обучающийся в целом хорошо представляет, каким образом функционирует микропроцессорная система и как организуется взаимодействие технических и программных средств, то можно полагать, что обучение было успешным.

Предметом настоящей книги является детальное изложение процесса работы микропроцессорной системы. В общем курсе по микропроцессорам обычно делают ссылки на многие частные аспекты этого процесса, предпочитая, как правило, их не описывать. В книге обсуждаются реальные системы, построенные на базе микропроцессоров. По мере ознакомления с конкретной системой подробно разбирается взаимодействие ее компонент. Рассматриваются схожие системы с вариацией лишь основной компоненты — микропроцессора, который выбирается из четырех основных типов — 8080, 8085, Z80, 6800. При этом система модифицируется, чтобы учесть частные особенности выбранного микропроцессора, а ее основные функции остаются неизменными. Это позволяет провести сравнение отличий и сходных черт различных микропроцессоров.

Основной акцент в книге сделан на технические средства. Программное обеспечение обсуждается лишь в той мере, в какой это необходимо для освещения вопросов управления техническими средствами. Приемы программирования здесь не затрагиваются, поскольку составляют предмет отдельного изложения.

Основная цель книги заключается в том, чтобы помочь читателю в использовании описанных микропроцессоров при построении и отладке технических средств системы в целом.

Все описанные в книге системы были реализованы и испытаны. Они используются на практике. Применяемые интегральные схемы являются стандартными изделиями, серийно выпускаемыми промышленностью, что обеспечивает возможность их приобретения, проверки и использования.

Определенное место отводится изложению вопросов отладки различных микропроцессорных систем. Это необходимо, чтобы убедиться, что технические средства функционируют в соответствии с заложенными требованиями. Для поиска неисправностей применяется методика, именуемая «Тестирование посредством статических сигналов». Также описана несложная техника и

средства отладки, которые могут быть использованы для большинства систем.

Необходимая справочная информация по интегральным схемам (ИС), используемым в различных устройствах, взята из документации фирм-изготовителей, так что читатель имеет все данные для проектирования. В книге описываются реальные схемы с реальными характеристиками. Выводы всех ИС промаркированы, а каждая схема детально описана.

Характеристики рассматриваемых схем рассчитываются, что позволяет читателю глубже понять возможности применения микропроцессоров, по мере ознакомления с которыми появляются настоящий интерес и желание реализовать полученные знания. Практическое воплощение при этом ограничивается лишь способностями и воображением проектировщика либо наличием ресурсов.

В будущем, возможно, появятся новые сферы применения микропроцессоров, о которых сегодня и не мечтают.

Дж. Коффон

ВВЕДЕНИЕ В АРХИТЕКТУРУ МАШИН С 3 ШИНАМИ

В настоящей главе мы обсудим общую архитектуру микропроцессорных систем. Рассматриваемые системы представляют особый интерес для тех, кто только приступает к изучению микропроцессоров. После общего описания организации микропроцессорной системы мы покажем особенности реализации типичной системы путем использования различных микропроцессоров: 8080, 8085, Z80 и 6800.

По мере совершенствования технологии постепенно вырабатываются предпочтительные способы исполнения изделий. При этом на этапах проектирования, производства, испытаний, отладки, установки и использования изделий применяются более или менее стандартные методы. То же относится и к микропроцессорам. Несмотря на относительную молодость микропроцессорной техники и на постоянно расширяющееся число микропроцессорных систем и их пользователей, в вопросах организации, структуры и архитектуры систем, в методах обработки данных, в уровнях логических сигналов, в анализе отказов и в интерфейсе с устройствами ввода-вывода уже выработался ряд стандартов. Поэтому назрела необходимость в одной книге рассмотреть устройства 8080, 8085, Z80, 6800. Начиная с описания единой архитектуры с 3 шинами, общие черты систем, построенных на базе указанных микропроцессоров, последовательно обсуждаются во всех главах.

При рассмотрении реальных систем с микропроцессорами мы опишем лишь те специальные интегральные схемы (ИС), предназначенные для использования с конкретными микропроцессорами, которые необходимы для понимания общих принципов работы системы. По мере возможностей будут использоваться обычные широко распространенные ИС. Если мы будем понимать, каким образом работает микропроцессорная система, построенная с помощью стандартных логических блоков, реализация этой системы со специальными логическими блоками не составит большого труда.

1.1. Архитектура систем с 3 шинами

Архитектура с 3 шинами является наиболее общей для микропроцессорных систем. Шинной системы называют физическую группу линий передачи сигналов, имеющих схожие функции в

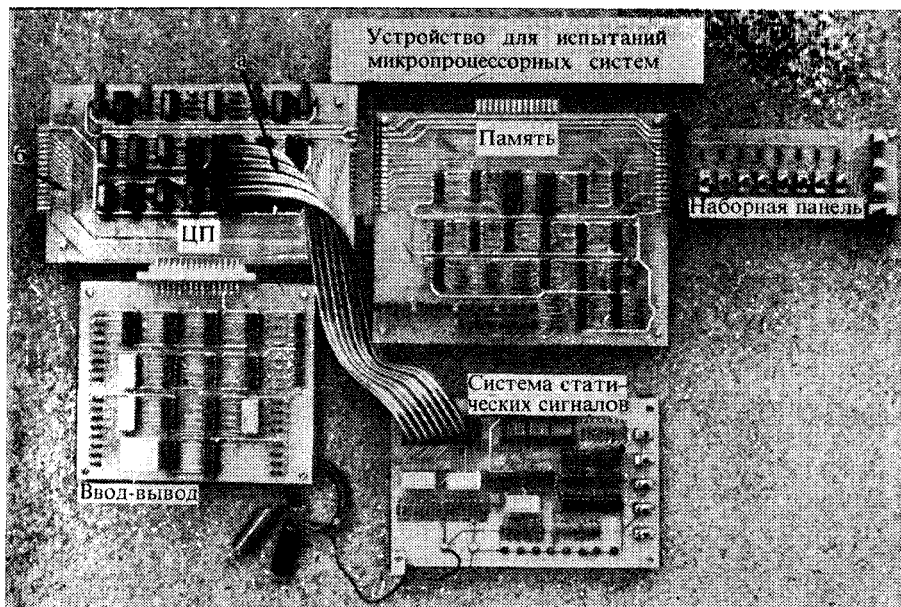


Рис. 1.1. Два примера физической реализации шины системы.

а — шина специального исполнения, состоящая из гибких проводов; *б* — шина, выполненная в виде печатной схемы.

рамках системы. Например, некоторая группа линий может использоваться для передачи сигналов адреса памяти. Эту группу линий можно назвать адресной шиной. На рис. 1.1 показаны два примера различной физической реализации шин. Все три шины являются специализированными с точки зрения их функций. Эти шины именуются так:

1. Адресная шина системы.
2. Шина данных системы.
3. Шина управления системы.

Логическое состояние этих трех шин описывает коммуникационный тракт системы в любой момент времени. Коммуникационный тракт — это путь, который данные, представленные в виде электрических сигналов, проходят в системе от одной точки к другой.

1.2. Адресная шина системы

По адресной шине системы передаются лишь выходные сигналы, которые поступают с выводов в корпусе микропроцессора. Эта шина предназначена для того, чтобы открывать или выби-

рать правильный тракт для электрического соединения в пределах микропроцессорной системы.

Для удобства будем в дальнейшем полагать, что все электрические соединения в микропроцессорной системе осуществляются между микропроцессором и устройством, открытым с помощью адресной шины. В качестве устройства здесь выступает любая электрическая схема, принимающая данные от микропроцессора либо вырабатывающая данные для него. (Позднее мы покажем, что это не всегда справедливо для систем, управляемых микропроцессором.) Для начинающих изучение микропроцессорных систем подобное ограничение способствует лучшему пониманию материала. После того как понятие описываемого здесь коммуникационного тракта дано, легче объяснить особенности других коммуникационных трактов, имеющих в микропроцессорной системе.

Другой важной характеристикой адресной шины системы является ее емкость. Емкость шины определяется числом входящих в нее отдельных электрических линий. Для микропроцессоров 8080, 8085, Z80, 6800 характерна 16-разрядная адресная шина. Это означает, что адресная шина систем, построенных на базе этих микропроцессоров, компонуется из 16 физических линий.

1.3. Шина данных системы

Шина данных системы является двунаправленной шиной. Это означает, что передача данных может производиться в обоих направлениях. В некоторых случаях данные генерируются микропроцессором и передаются от него к определенному устройству системы. Это устройство открывается с помощью заданного логического состояния линий адресной шины и получает данные с шины данных.

В других случаях данные генерируются каким-то источником и передаются микропроцессору посредством шины данных. В качестве источника выступает то устройство системы, которое открывается с помощью адресной шины. Подобный режим называется вводом данных в микропроцессор.

Хотя передача данных по шине данных может производиться в обоих направлениях, однако в каждый заданный момент времени она осуществляется лишь в одном направлении. Это означает, что для передачи данных в систему и их приема из системы микропроцессор переводится в соответствующий режим. Более того, по всем разрядам шины в каждый момент времени данные передаются лишь в одном направлении, т. е. в любой момент по всем линиям шины они могут либо только вводиться либо только выводиться.

Для микропроцессоров 8080, 8085, Z80 и 6800 шина данных является 8-разрядной. Поэтому говорят, что емкость шины дан-

ных равна 8 разрядам и параллельно могут передаваться лишь 8 бит информации. По этой причине перечисленные микропроцессоры относят к классу 8-разрядных микропроцессоров.

1.4. Шина управления системы

На шине управления действует 4 следующих типа сигналов:

1. Чтение из памяти активизировано.
2. Запись в память активизирована.
3. Чтение с устройства ввода активизировано.
4. Запись на устройство ввода активизирована.

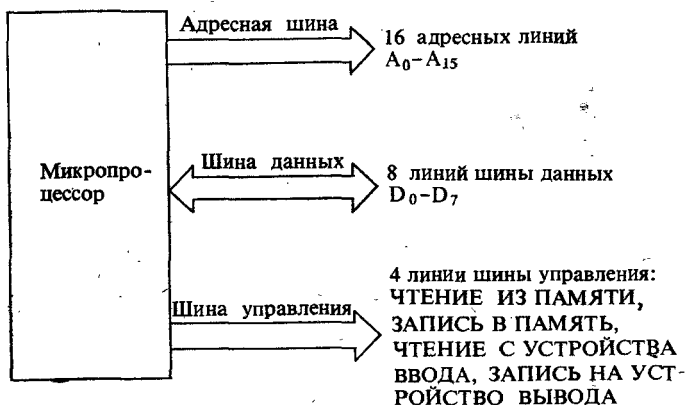


Рис. 1.2. Блок-схема системы с 3 шинами. По адресной шине и шине управления сигналы передаются лишь в одном направлении (однонаправленные шины). Шина данных является двунаправленной, что показано стрелкой, имеющей два направления.

Позднее для этой шины мы введем некоторые добавочные сигналы. Однако для понимания существа процессов пока необходимо ограничиться указанным списком сигналов. После того как станут ясны функции этих четырех сигналов, будет легче изучать функции других сигналов.

Шина управления используется лишь для вывода сигналов, т. е. является однонаправленной и работает лишь в режиме вывода. Напротив, шину данных мы рассматриваем как двунаправленную.

Термин «активизирован» означает, что при наступлении события, запрашиваемого соответствующей линией шины управления, эта линия имеет активный сигнал логического уровня 1 или 0. В микропроцессорных системах активное состояние линий шины управления может быть либо логической 1, либо ло-

гическим 0. При этом различные линии системы могут быть активными при разных уровнях логического сигнала. Например, линия управления ЧТЕНИЕ ИЗ ПАМЯТИ может быть активной при логическом уровне 1, а линия управления ЗАПИСЬ В ПАМЯТЬ — при уровне 0.

Распознавание и инициирование типа электрического соединения для шины данных системы является функцией сигнала шины управления. Необходимо заметить, что линия управления может быть активизированной и при уровне логической 1, и при уровне логического 0.

На рис. 1.2 архитектура систем с тремя шинами показана в виде блок-схемы, из которой видно, что стрелки, соответствующие адресной шине и шине управления, указывают лишь на одно направление. Это говорит о том, что эти шины однонаправленные. Для шины данных на рис. 1.2 стрелки указывают на два направления, что соответствует двунаправленной шине. Принятые нами обозначения часто используются в литературе для описания этих шин.

1.5. Использование архитектуры с 3 шинами

Теперь мы обсудим общие принципы передачи информации в микропроцессорной системе, имеющей архитектуру с 3 шинами. Прежде всего необходимо пояснить основные функции, реализуемые микропроцессорной системой. После этого можно будет перейти к рассмотрению особенностей их выполнения.

Для первоначального знакомства с микропроцессорами достаточно рассмотреть лишь пять функций, описанных ниже; позднее список этих функций может быть расширен. Такие функции хорошо отображают возможные операции, выполняемые в микропроцессорной системе. К ним относятся:

1. Запись данных в память системы.
2. Чтение данных из памяти системы.
3. Запись данных в устройство ввода-вывода.
4. Чтение данных с устройства ввода-вывода.
5. Выполнение операций с содержимым внутренних регистров микропроцессора.

Указанные пять возможных типов функций микропроцессорной системы позволяют создавать большое число разнообразных средств. Рассмотрим, каким образом эти функции могут быть реализованы посредством архитектуры систем с 3 шинами.

1.6. Запись данных в память

Чтобы понять, каким образом в микропроцессорной системе осуществляется запись данных в память, необходимо выяснить особенности передачи данных в память от любого внешнего источ-

ника. С этой целью мы приведем временную диаграмму общего процесса записи данных в полупроводниковую память¹⁾, после рассмотрения которой будет легко показать особенности использования системы с 3 шинами для выполнения этой операции.

Последующее обсуждение относится к рис. 1.3. Отметим, что адресные входы памяти на рисунке маркируются как A_0 — A_n ,

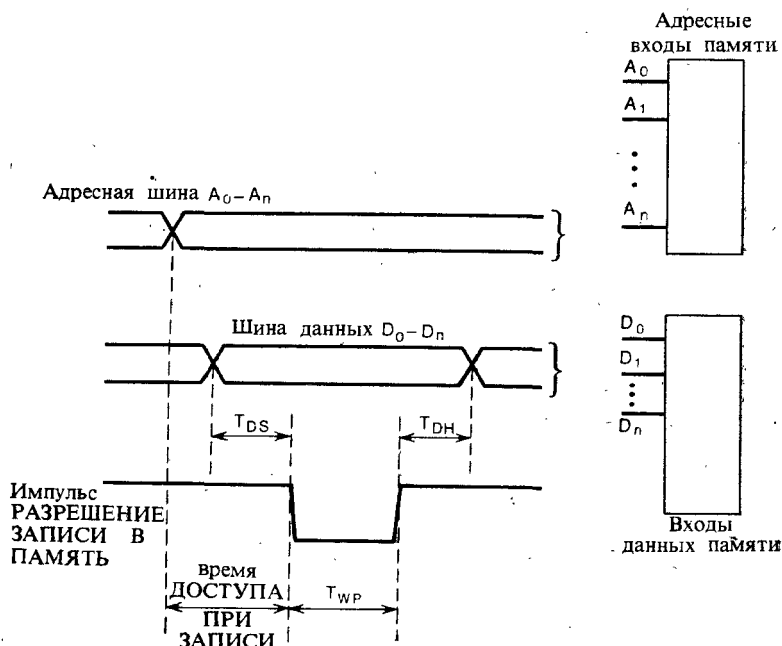


Рис. 1.3. Упрощенная временная диаграмма трех сигналов, необходимых для реализации цикла ЗАПИСЬ В ПАМЯТЬ для полупроводниковой памяти.

где A_n характеризует максимальное число адресных линий, необходимых для данного блока памяти. Например, если память организована как 1024×1 , то для обеспечения доступа к любой ячейке памяти необходимо 10 линий, и адресные линии будут обозначаться как A_0 — A_9 .

Линии данных маркируются как D_0 — D_n , где D_n характеризует максимальное число линий данных памяти. Например, если память организована как 256×4 бит, необходимо четыре линии данных. Линии данных такой памяти обозначаются D_0 — D_3 .

Заметим также, что как адресные входы, так и входы данных должны быть активизированы после выдачи сигнала раз-

¹⁾ Для углубленного изучения функционирования полупроводниковой памяти см. Coffron J. W., Getting Started in Digital Troubleshooting, Reston Publishing Co., Reston, VA, 1979.

решения записи. Термин «активизировать» здесь означает подачу на адресные линии и линии данных уровня напряжения, соответствующего логическим 1 или 0 и удовлетворяющего принятым для используемого семейства логических схем параметрам. Например, если в системе используются схемы семейства TTL, то активизация логической 1 соответствует интервалу 2,4—5,0 В, а логического 0—0,0—0,4 В.

Интервал времени, в течение которого должны сохраняться активизированными адресные и информационные входы до выдачи сигнала разрешения записи, для различных устройств разный. Например, для устройств памяти типа МОП (металл — окисел — полупроводник) этот интервал составляет 200 нс, а для памяти на схемах TTL — 30 нс. Для точного определения временных соотношений следует ознакомиться с технической документацией на соответствующие изделия.

Анализируя рис. 1.3, важно подчеркнуть, что вне зависимости от конкретных временных соотношений сигналов для заданного устройства памяти микропроцессорная система должна удовлетворять всем существующим временным параметрам для устройства памяти данного типа. Если же эти временные соотношения не соблюдаются, то надежной взаимосвязи между микропроцессором и памятью системы обеспечить не удастся.

Для того чтобы успешно реализовать обработку сигнала записи в память в системе с 3 шинами, на шинах системы необходимо выполнить следующие действия:

1. На адресной шине A_0 — A_{15} должен быть активизирован адрес памяти (т. е. адрес ячейки, куда записываются данные, генерируемые микропроцессором, с шины данных).

2. На шину данных D_0 — D_7 должны поступить данные из микропроцессора. (Эти данные необходимо записать в ячейку, адрес которой содержится на адресной шине.)

3. После осуществления действий 1 и 2 на линию записи в память шины управления должен поступить соответствующим образом синхронизированный импульс разрешения записи требуемого уровня напряжения. При этом осуществляется передача данных. Таким образом, шина управления производит управление системой посредством соответствующим образом синхронизированных импульсов.

Однако существует ряд дополнительных действий, которым также необходимо уделить внимание. Об этих действиях будет говориться ниже при реализации систем с 3 шинами на базе конкретных микропроцессоров. Схемы, реализующие их, проще описывать при более детальном ознакомлении с системой. В настоящей главе раскрываются лишь общие вопросы функционирования микропроцессоров.

При обсуждении мы намеренно опускаем частные детали. Это объясняется стремлением пояснить общие принципы функ-

ционирования микропроцессорных систем с 3 шинами прежде, чем рассматривать реальные системы.

Из обсуждения рис. 1.3 ясно, каким образом сигналы различного логического уровня на каждой из шин (адресной, данных, управления) взаимодействуют между собой при формировании необходимой входной информации при записи данных в память системы. Шины системы обеспечивают надежное соединение микропроцессора с памятью системы путем формирования необходимой комбинации сигналов на входах памяти.

1.7. Чтение данных из памяти

Покажем, каким образом в системе с 3 шинами реализуется чтение данных из памяти. Кратко рассмотрим основные особенности чтения данных из любой полупроводниковой памяти. На

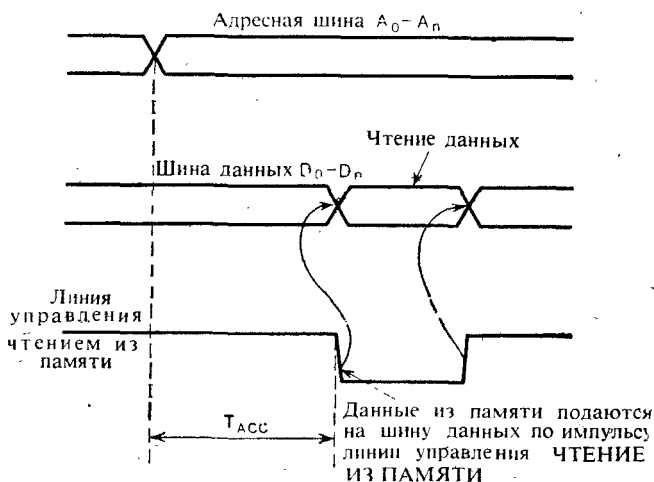


Рис. 1.4. Типичная временная диаграмма цикла ЧТЕНИЕ ИЗ ПАМЯТИ.

рис. 1.4 приведена типичная временная диаграмма выполнения операции чтения данных из памяти. Отметим, что в соответствии с рис. 1.4 адресные линии $A_0 - A_n$ должны быть активизированы с сохранением стабильного уровня сигнала до момента времени, когда данные из памяти поступают на шину данных по сигналу соответствующей линии шины управления. Вспомним, что данные в системе передаются из одного устройства в другое посредством шины данных. Для выполнения этой операции шина данных переводится в такой режим, при котором логические уровни на этой шине соответствуют данным, хранимым в памяти. И наконец, сигнал шины управления определяет нужный момент выдачи данных из памяти на шину данных. Таким

образом, чтобы реализовать операцию чтения данных из памяти системы, необходимо выполнить следующие действия:

1. Обеспечение стабильных уровней сигналов на адресной шине A_0 — A_n .

2. Подготовка шины данных для приема данных (т. е. она переводится в режим приема данных в микропроцессор).

3. После реализации шагов 1 и 2 активизация шиной управления линии управления чтением из памяти. При этом данные из памяти поступают на шину данных и могут быть восприняты микропроцессором. Как видим, реализация операций чтения из памяти предполагает взаимодействие сигналов соответствующих логических уровней на всех трех шинах.

1.8. Запись данных в устройство вывода

В микропроцессорной системе слово «память» имеет вполне определенное значение, а именно оно ассоциируется с некоторой схемой или множеством схем, с которыми можно легко связать функции чтения и записи. Иначе обстоит дело с понятием «устройство вывода», оно определено не достаточно четко. Устройством вывода может быть и построчное печатающее устройство, и табло на светоизлучающих диодах, и электронно-лучевая трубка (ЭЛТ), и кассетный накопитель, и гибкий диск и любое другое устройство микропроцессорной системы. В интересах последующего изложения определим устройство вывода следующим образом:

«Устройство вывода — это любое устройство, кроме памяти, воспринимающее данные, передаваемые от микропроцессора».

Такой подход к определению устройства вывода позволяет представлять его в виде устройства, способного принимать от 1 до 8 бит в параллельном режиме. Причем прием данных осуществляется в том случае, если на шине управления активизирована линия управления записью на устройство вывода.

Функционирование устройства вывода, так же как и функционирование памяти, можно рассмотреть с позиции взаимодействия электрических сигналов. Отличие заключается лишь в том, что устройство вывода осуществляет обработку сигнала управления, а память системы — нет. Так микропроцессор 6800 одинаковым образом производит запись данных в память и в устройство вывода. Это справедливо для многих микропроцессоров. Системы, в которых реализуется подобное взаимодействие, называют системами с организацией вывода по аналогии с обращением к памяти. Изложенное должно помочь начинающим изучение микропроцессорных систем понять, что отличие между организацией обмена с памятью и с устройствами вывода в таких системах невелико.

Каждому устройству вывода приписан свой адрес, подобный адресу памяти системы. На рис. 1.5 показана в упрощенном виде временная диаграмма записи в устройство вывода. Из рис. 1.5 важно понять, что адрес устройства вывода и выводимые данные поступают на соответствующие шины до того, как шиной управления системы будет выдан импульс разрешения вывода. Операция записи в устройство вывода выполняется путем реализации следующей последовательности действий:

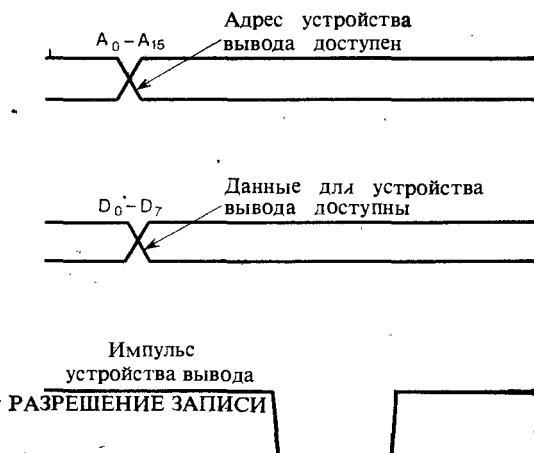


Рис. 1.5. Упрощенная временная диаграмма операции ЗАПИСЬ В УСТРОЙСТВО ВЫВОДА для системы с 3 шинами.

1. Обеспечение стабильных уровней сигналов A_0-A_{15} , соответствующих адресу устройства вывода. В некоторых системах для адресации устройства вывода используются не все 16 адресных линий. Более подробно этот вопрос будет изложен позднее, когда будет рассматриваться система с 3 шинами на базе реального микропроцессора.

2. Обеспечение на шине данных D_0-D_7 стабильных уровней сигналов, соответствующих данным, которые должны быть записаны в устройство вывода.

3. После выполнения шагов 1 и 2 подача на шину управления сигнала разрешения записи в устройство вывода.

Рассмотренная выше операция записи данных в устройство вывода выявляет функцию каждой шины системы. При реализации этой операции все три шины функционируют во взаимодействии друг с другом. И используя их, система устанавливает надежную связь между микропроцессором и устройством вывода.

1.9. Чтение данных с устройства ввода

В качестве устройства ввода микропроцессорной системы может рассматриваться любой источник, кроме памяти системы, способный передавать данные микропроцессору. Данные из устройства ввода пересылаются в микропроцессор по запросу последнего. В качестве примера устройства ввода можно назвать клавишный пульт терминала с ЭЛТ. Данные с пульта поступают в микропроцессор по его запросам.

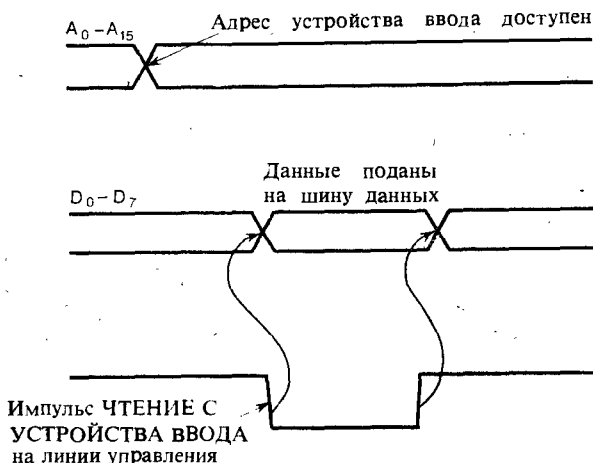


Рис. 1.6. Временная диаграмма цикла ЧТЕНИЕ С УСТРОЙСТВА ВВОДА, показывающая основные сигналы шины.

В некоторых системах микропроцессор не делает различия между чтением с устройства ввода и из памяти системы. Подобное замечание ранее было сделано относительно устройства вывода. Если микропроцессорная система не различает чтение данных из памяти и с устройства ввода, то говорят, что в системе используется ввод данных по аналогии с обращением к памяти. Архитектура системы подобного типа упоминалась в связи с записью данных на устройство вывода.

Наиболее типичным примером системы, в которой ввод-вывод данных реализуется по аналогии с обращением к памяти, является микропроцессор 6800. Более подробно этот вопрос будет изложен в последующих главах при рассмотрении принципов построения систем с 3 шинами на базе устройства 6800.

На рис. 1.6 представлена временная диаграмма, показывающая взаимосвязь между тремя шинами. Из этого рисунка видно, что адрес устройства ввода поступает на адресную шину до того, как на шину управления подается сигнал управления чтением.

ем с устройства ввода. В течение промежутка времени, пока на шине управления сохраняется этот сигнал, данные с устройства ввода передаются на шину данных.

Сравнивая рассмотренные до сих пор временные диаграммы, можно заметить, что синхронизация процессов при чтении данных из памяти и с устройства ввода схожа. То же самое относится и к синхронизации процессов при записи данных в память и на устройство вывода. Как уже упоминалось, с помощью большинства микропроцессоров можно строить системы, в которых нет различия между записью данных в память и на устройство вывода либо чтением данных из памяти и с устройства ввода.

1.10. Операции с внутренними регистрами

Операции с внутренними регистрами представляют собой пятую функцию из ряда основных, реализуемых микропроцессорной системой. Примером такой операции может служить операция ADD, по которой содержимое внутреннего регистра микропроцессора складывается с содержимым аккумулятора. Для разных микропроцессоров существуют особенности выполнения этой операции, однако при этом все необходимые действия реализуются внутри «единственного кристалла» микропроцессора, так что внешние шины системы не используются.

В начале нашего обсуждения мы установили, что пять перечисленных функций микропроцессора позволят начинающему читателю адекватным образом описать большинство процессов в микропроцессорной системе. Позднее мы обсудим некоторые дополнительные функции, а пока основное внимание будет сосредоточено на пяти указанных, которые постоянно используются вне зависимости от сложности программного обеспечения системы. В качестве примера рассмотрим в общем виде выполнение команд в микропроцессорной системе. После этого станет очевидной периодическая повторяемость пяти основных функций.

1.11. Выполнение команд в системе с 3 шинами

Основная последовательность действий при выполнении любой команды такова:

1. Микропроцессор выдает в память адрес, по которому хранится код операции команды.
2. Код операции читается из памяти и вводится в микропроцессор (функция 2).
3. Команда дешифрируется микропроцессором.
4. Микропроцессор «настраивается» на выполнение одной из пяти основных функций в соответствии с результатами дешифрирования считанного кода операции.

5. Предположим, что была считана команда с непосредственными данными. По этой команде в один из внутренних регистров микропроцессора загружается константа. Микропроцессор настраивается на чтение второго байта данных из памяти.

6. Второй байт данных из памяти загружается во внутренний регистр микропроцессора (функция 5).

Главное здесь заключается в том, что вне зависимости от кода операции микропроцессор настраивается на выполнение одной из пяти рассмотренных выше функций.

Снова подчеркнем, что существуют и другие реализуемые системой функции, например прерывание, прямой доступ к памяти (ПДП), ожидание. Эти функции будут обсуждаться в последующих главах. Однако вполне возможно спроектировать управляемую микропроцессором систему, в которой не используются никакие дополнительные функции, за исключением пяти приведенных выше. Позднее рассмотрим проектирование таких систем. Однако чтобы не оказаться во власти заблуждения, следует помнить, что приведенный здесь подход к проектированию имеет не только методологическое значение и оказывается весьма эффективным при настройке, проектировании и отладке многих систем, управляемых микропроцессорами.

1.12. Управление синхронизацией системы

При изучении различных временных соотношений, соблюдение которых необходимо для реализации пяти основных функций, возникает естественный вопрос: каким образом осуществлять правильную синхронизацию всех сигналов? Фундаментальная особенность использования микропроцессоров при проектировании систем заключается в следующем: синхронизация всех сигналов в системе осуществляется схемами, входящими в состав кристалла микропроцессора.

При обработке микропроцессором программной команды чтения из памяти выработка и синхронизация необходимых сигналов для формирования адреса, для выборки данных и для управления осуществляются самим микропроцессором.

Пользователи микропроцессора должны знать синхронизацию всех его сигналов. Это способствует правильному использованию временных соотношений при проектировании системы. Такой несколько упрощенный подход позволяет достаточно точно представить функционирование большого числа существующих микропроцессоров.

В дальнейшем этот подход будет использован при проектировании микропроцессорных систем на базе устройств 8080, 8085, Z80, 6800.

1.13. Выводы

В настоящей главе была рассмотрена архитектура типичной микропроцессорной системы с 3 шинами. Эти три шины включают: адресную шину; шину данных; шину управления.

При реализации общих системных функций было также показано взаимодействие сигналов различного уровня, подаваемых на эти шины. К этим функциям относятся:

1. Запись данных в память системы.
2. Чтение данных из памяти системы.
3. Запись данных в устройство вывода системы.
4. Чтение данных с устройства ввода системы.
5. Выполнение операций с внутренними регистрами.

Было показано, каким образом эти основные функции реализуются в системе с 3 шинами.

Основные положения настоящей главы применимы к весьма широкому кругу микропроцессорных систем. В последующих главах эти общие положения будут использованы при построении реальных микропроцессорных систем. В гл. 2 определяется архитектура систем с 3 шинами на базе существующих схем микропроцессоров 8080, 8085, Z80 и 6800. В проектируемых устройствах рассматриваемые в главе общие положения приобретают практический смысл.

Глава 2

ПОСТРОЕНИЕ СИСТЕМ С 3 ШИНАМИ НА БАЗЕ УСТРОЙСТВ 8080, 8085, Z80 и 6800

В настоящей главе будет рассмотрено, каким образом архитектура систем с 3 шинами может быть реализована с использованием реальных микропроцессоров. Одновременно будут обсуждаться микропроцессоры 8080, 8085, Z80 и 6800, для каждого из которых будут спроектированы шины адресная, данных и управления.

В процессе проектирования этих шин будут рассмотрены особенности реализации шин для каждого микропроцессора. В главе приводятся реальные схемы. Они могут быть использованы в качестве центрального процессора (ЦП), построенного на любом из рассматриваемых микропроцессоров. После построения шин будут обсуждены вопросы интегрирования памяти, устройств ввода и вывода в рамках одной микропроцессорной системы. Эти вопросы поднимаются в настоящей и последующих главах.

2.1. Пояснения к адресной шине системы

При проектировании адресной шины прежде всего оценивают величину токовой нагрузки, при которой каждая адресная шина работает удовлетворительно. Адресные линии в микропроцессорной системе связаны со множеством адресных входов, подключенных параллельно, как это показано на рис. 2.1. Если для адресной линии характерен ток, по величине превосходящий допустимое значение на выходе микропроцессора, то такую линию необходимо буферизировать.

Под буферизированием понимают такую процедуру, при которой адресная линия снабжается средствами «съема» избыточного тока. Например, предположим, что адресные линии микропроцессорной системы должны управлять адресными входами четырех блоков памяти и двух портов ввода-вывода, что показано на рис. 2.2. Не будем обсуждать правомерность приведенных в схеме соединений. Этот вопрос раскрывается в гл. 3. Просто допустим, что приведенные соединения возможны.

Величина тока в адресной линии, управляющей блоками памяти и устройствами ввода-вывода, равна сумме входных токов всех четырех блоков памяти и двух портов ввода-вывода. Пусть

в качестве памяти используется оперативное запоминающее устройство (ОЗУ) 1024×4 бит типа 2114.

По техническим характеристикам устройства 2114 видно, что нагрузка каждого входа равна 10 мкА и не зависит от логического состояния адресной линии. Это означает, что общая нагрузка от микропроцессора на адресной линии при работе с памятью есть 4×10 или 40 мкА.

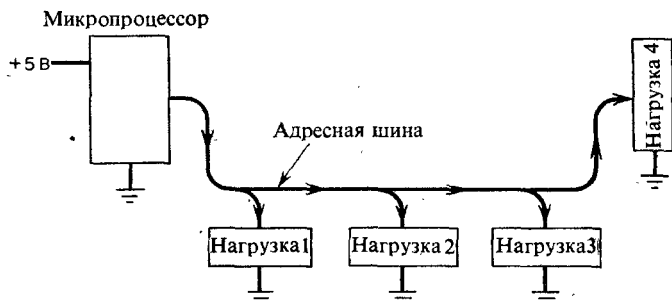


Рис. 2.1. Подсоединение к адресной шине микропроцессора нескольких устройств (нагрузок). На схеме показаны четыре нагрузки.

Далее допустим, что при построении входных портов рис. 2.2 используются устройства 74LS (маломощные устройства Шоттки). Это означает, что каждое устройство вывода системы создает на адресной линии нагрузку, соответствующую стандартной для 74LS. Согласно техническим характеристикам 74LS, эта нагрузка равна 20 мкА при сигнале логической 1 и 0,4 мА — при логическом 0. В табл. 2.1 приведены значения суммарной нагрузки на адресной линии системы, показанной на рис. 2.2. Из табл. 2.1 видно, что максимальная нагрузка на адресной линии для состояния логической 1 равна 80 мкА, а для состояния логического 0 — 840 мкА.

Таблица 2.1

Примерная нагрузка для адресной линии A_0 (рис. 2.2)

Тип памяти	Нагрузка (мкА)	
	Для одного блока	Для всей памяти
2114	10 мкА (логическая 1)	40 мкА (логическая 1)
	10 мкА (логический 0)	40 мкА (логический 0)
Тип устройства ввода-вывода 74LSXX	Для каждого устройства	Для всех устройств
	400 мкА (логический 0)	800 мкА (логический 0)
	20 мкА (логическая 1)	40 мкА (логическая 1)
Общая нагрузка		
	40 мкА + 800 мкА = 840 мкА (логический 0)	
	40 мкА + 40 мкА = 80 мкА (логическая 1)	

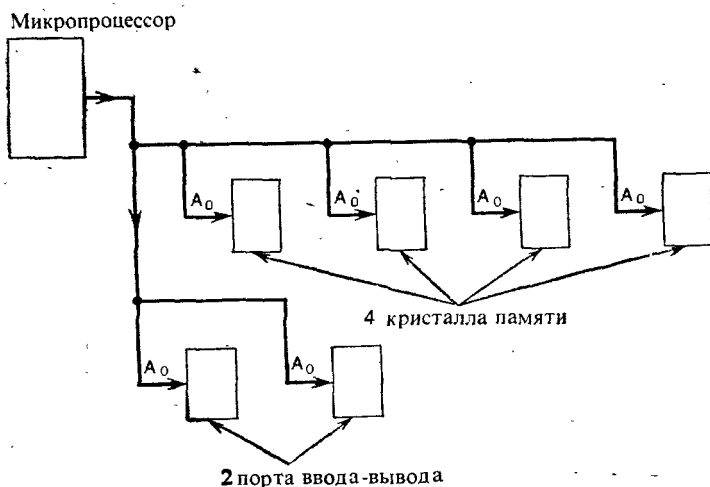


Рис. 2.2. Использование адресной линии A_0 , исходящей из микропроцессора, для управления четырьмя линиями памяти и двумя портами ввода-вывода (заземление на схеме не показано).

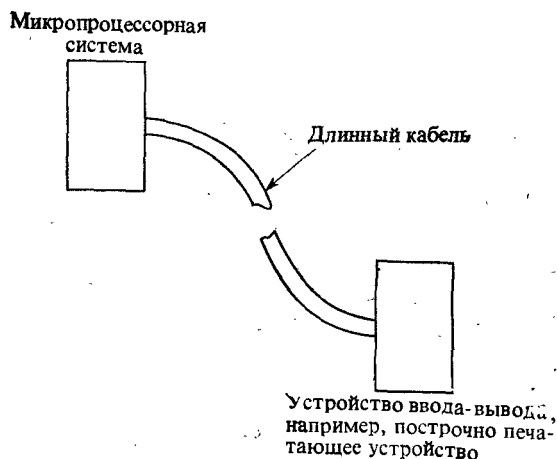


Рис. 2.3. Микропроцессорная система может использоваться с устройством вывода типа построчно печатающего устройства, которое может соединяться с системой посредством длинного кабеля. Такой кабель обладает большой емкостью, поэтому для передачи по нему сигнала необходимо использовать специальные схемы.

После того как на адресной линии приблизительно оценена максимальная нагрузка, сравним ее с допустимой, определяемой характеристиками используемого микропроцессора. Например, для микропроцессора 8080 максимальная допустимая нагрузка на адресной линии в состоянии логического 0 есть 1,9 мА, а в состоянии логической 1—150 мкА. Сравнивая эти данные, видим, что в обоих случаях (при логических 1 и 0) рассчитанная нагрузка меньше допустимой.

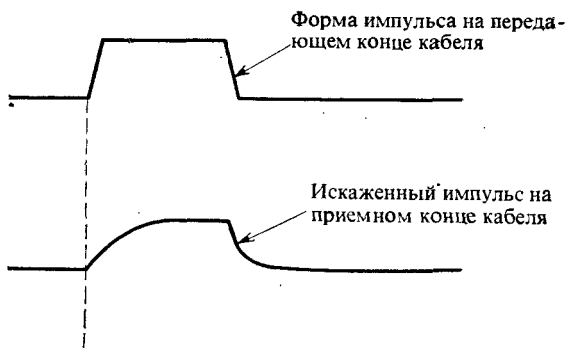


Рис. 2.4. Небуферизованный сигнал, передаваемый по длинному кабелю, может искажаться. Искажение может оказаться настолько значительным, что на приемном конце принимается ложный сигнал.

В приведенном примере микропроцессор в состоянии непосредственно управлять памятью и устройствами вывода. В более мощных системах он осуществить такое управление не сможет. Если нагрузка по току на адресных линиях превышает его возможности, возникает необходимость введения буферов. Однако при этом необходимо рассмотреть ряд дополнительных факторов.

Введение буфера адреса требует тот факт, что адресные линии рассчитывались на большую емкостную нагрузку. Примером емкостной нагрузки является кабель очень большой длины, иными словами — устройство вывода, подсоединенное к системе с помощью длинного кабеля. Подобная ситуация показана на рис. 2.3. Здесь для поддержания электрических характеристик сигнала перед его передачей по кабелю для адресных линий необходим буфер. Причиной этого служит большая емкость кабеля. При ограниченных возможностях по формированию тока, как показано на рис. 2.4, сигнал, получаемый на приемном конце кабеля, будет искажен.

Другим примером, где, возможно, необходимо буферирование адреса, является система, в рамках которой адресные линии соединяются с различными платами. При этом микропроцессор монтируется на одной плате, а адресные линии проходят через

торцевой разъем и соединяются с другой платой в пределах одной системы. Подобная ситуация иллюстрируется схемой рис. 2.5. Такая система характеризуется очень высокой емкостной нагрузкой на отдельные линии, проходящих через торцевые разъемы и заднюю поверхность основной платы.

В случае когда целесообразность буферирования сомнительна, полезно помнить простые правила. Используйте буферы адреса:

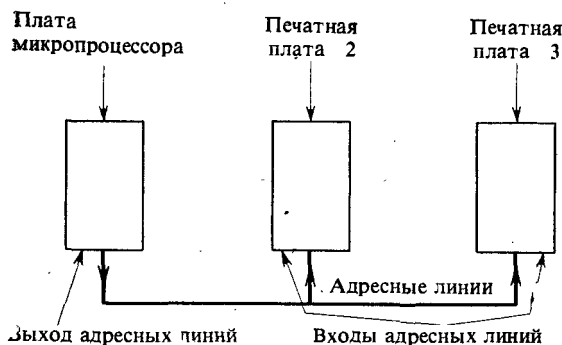


Рис. 2.5. Блок-схема типичной системы, в которой адресные линии, начинаясь на одной печатной схеме, проходят на другую в пределах системы. Здесь необходимо использовать буферы адреса на плате, откуда исходят адресные линии.

1) если нагрузка по току для любой адресной линии применяемого микропроцессора приблизительно равна или больше номинальной.

2) если адресные линии соединены с какой-либо схемой, расположенной вне платы, на которой генерируются соответствующие сигналы.

2.2. Выбор требуемых буферов адреса

После решения вопроса о необходимости буферирования следует выбрать подходящие буферы. При выборе буферов необходимо проанализировать большое число факторов. При этом многие проектировщики имеют склонность к тем или иным устройствам. Ниже приводятся правила, которые могут помочь в решении этой задачи.

1. Используйте неинвертирующий буфер. Хотя это может показаться очевидным, тем не менее начинающие работать с микропроцессорными системами, до тех пор пока буфер не включен в схему, часто не предполагают, что он инвертирующий. (Для некоторых микропроцессорных систем при проектировании предполагается использование инвертирующих буферов адреса, так

что это правило не может быть строго обязательным.) В общем случае начинающий облегчит себе задачу, если будет использовать неинвертирующий буфер.

2. Используйте тот буфер, который по возможности обеспечивает все требуемые в системе значения выходной нагрузки. Если, например, в системе необходима выходная нагрузка в 18 мА на адресной линии, то используемый буфер должен обеспечить нагрузку 18 мА или более. Это будет способствовать снижению числа выводов в корпусе устройства. Если же используется буфер, обеспечивающий лишь часть требуемой нагрузки, то для реализации оставшейся нагрузки потребуются еще один буфер.

3. Убедитесь, что нагрузка по входу для выбранного буфера не превышает выходную нагрузку адресной линии микропроцессора. Например, если величина тока на входе буфера адреса при состоянии логического 0 составляет 2 мА, то микропроцессор не сможет управлять этим буфером, если его выходной ток менее 2 мА.

Позднее мы расширим этот набор правил. В качестве буферов адреса могут быть использованы, например, следующие устройства:

1. 7407 (схема со свободным коллектором).
2. 74LS367 шестивходовый буфер с тремя состояниями.
3. 74LS125 четырехходовый буфер с тремя состояниями.

Кристаллы многих микропроцессоров имеют специальные интегральные схемы, предназначенные для буферирования адреса.

Ниже приводится пример, в котором в качестве буфера адреса будет использовано произвольно выбранное устройство 74LS367.

Буферирование адреса использовано для всех рассматриваемых адресных шин, хотя это не во всех случаях обязательно.

2.3. Адресная шина микропроцессора 8080

Рассмотрим проектирование адресной шины для микропроцессора 8080. Ниже будет показано назначение отдельных выводов и описаны конкретные устройства, используемые при построении буферов адреса (т. е. сделано допущение, что в системе необходимо буферирование адреса). Это позволит ознакомиться с использованием буферов адреса в реальной системе.

Приведенные на рис. 2.6 спецификации для адресных выводов микропроцессора 8080 показывают, что адресные выходы в состоянии логического 0 характеризуются током 1,9 мА, а в состоянии 1 — током 150 мкА. Допустим, что управляемая система имеет нагрузку в 3 мА при логическом 0 и 200 мкА — при логической 1. Эти параметры выбраны с целью задания условий, при которых адресные выходы микропроцессора требуют буферирования.

Характеристики по постоянному току

$T_A = 0-70^\circ\text{C}$, $V_{DD} = +12 \text{ В} \pm 5\%$, $V_{CC} = +5 \text{ В} \pm 5\%$, $V_{BB} = -5 \text{ В} \pm 5\%$, $V_{SS} = 0 \text{ В}$,
если не оговорено противное

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{ILC}	Низкий уровень тактового напряжения	$V_{SS}-1$		$V_{SS}+0,8$	В	
V_{IHC}	Высокий уровень тактового напряжения	9,0		$V_{DD}+1$	В	
V_{IL}	Низкий уровень входного напряжения	$V_{SS}-1$		$V_{SS}+0,8$	В	
V_{IH}	Высокий уровень входного напряжения	3,3		$V_{CC}+1$	В	
V_{OL}	Низкий уровень выходного напряжения			0,45	В	$I_{OI} = 1,9 \text{ мА}$ на всех выходах, $I_{OH} = 150 \text{ мкА}$
V_{OH}	Высокий уровень выходного напряжения	3,7			В	
I_{DD}	Средний ток при V_{DD} » » » V_{CC} » » » V_{BB} Ток утечки на входе Ток ут. на вх. синхронизации Ток ут. на шине данных в режиме ввода Ток ут. на адресной шине и шине данных в режиме HOLD		40	70	мА	Режим $T_{cy} = 0,32 \text{ мс}$ $V_{SS} \leq V_{IN} \leq V_{CC}$ $V_{SS} \leq V_{TAKT} \leq V_{DD}$ $V_{SS} \leq V_{IN} \leq V_{SS} + 0,8 \text{ В}$ $V_{SS} + 0,8 \text{ В} \leq V_{IN} \leq V_{CC}$ $V_{адрес/данные} = V_{CC}$ $V_{адрес/данные} = V_{SS} + 0,45 \text{ В}$
I_{CC}			60	80	мА	
I_{BB}			0,01	1	мА	
I_{IL}				± 10	мкА	
I_{CL}				± 10	мкА	
I_{DL}				-100	мкА	
I_{FL}				-2,0	мА	
				+10	мкА	
				-100	мкА	

Рис. 2.6. Отдельные спецификации для адресных выходов микропроцессора 8080. Наиболее важные из них выделены жирной рамкой.

2.4. Использование буферов адреса

В качестве буферов адреса выберем устройства 74LS367. Со спецификациями этих устройств можно ознакомиться по технической документации. Некоторые из этих спецификаций приведены на рис. 2.7, откуда видно, что потребляемый на входе 74LS367 ток равен 0,3 мА в состоянии логического 0 и 20 мА в состоянии логической 1. Ток такой величины может быть обеспечен на адресных выходах микропроцессора 8080, что иллюстрирует рис. 2.8.

Из спецификаций устройства 74LS367, приведенных на рис. 2.8, видно, что с выхода этого устройства поступает ток 2 мА в состоянии логической 1 и 12 мА в состоянии логического 0. Этого более чем достаточно для выбранного применения. На рис. 2.9 показана полная адресная шина микропроцессора 8080 с подсоединенными буферами 74LS367.

Из рис. 2.9 видно, что выводы 15 и 1 устройства 74LS367 заземлены непосредственно. Соединение такого рода характерно для многих микропроцессорных систем, что позволяет открывать буферы с тремя состояниями. Однако в некоторых ситуациях при прямом доступе к памяти может возникнуть необходимость в блокировании буфера с тремя состояниями. Особенности прямого доступа к памяти пока не раскрыты, и поэтому данный аспект адресования будет рассмотрен позднее. Пока достаточно лишь понять, что в случае, когда выводы 15 и 1 устройства 74LS367 заземлены, или находятся в состоянии логического 0, буферы с тремя состояниями разблокированы.

Далее из рис. 2.9 следует, что наименование сигналов меняется после прохождения адресных линий через буфер. При построении схемы системы очень важно, чтобы два различных сигнала никогда не назывались одинаковым именем. В приведенном примере адресные линии, исходящие из микропроцессора, обозначаются как A_0 — A_{15} , а адресные линии на выходе устройства 74LS367 — соответственно как BA_0 — BA_{15} . Это обозначение говорит о том, что адресные линии A_0 — A_{15} буферизованы. Однако важнее не выбор конкретного обозначения, а сам факт иного обозначения линий с различной величиной сигнала в пределах системы.

Важность хорошей документации трудно переоценить. Причина выбора единой всеобъемлющей системы обозначений становится ясной при проведении монтажа или поиска неисправностей. Монтаж системы всегда затруднен, если из обозначений не ясно, какая линия должна соединяться с заданной точкой. Небольшие начальные затраты времени на введение удобных обозначений сигналов позволят проектировщику и другим специалистам, связанным с разработкой системы, сэкономить много времени позднее.

Писать полевые шинами формирователи
367

Низкий уровень входных сигналов; 4х линейные и 2х линейные отпаса- макс. входные сигналы с 3

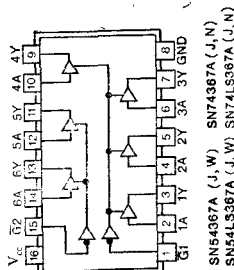


Рис. 2.7. Частные спецификаций шинных формирователей 74LS367. Наиболее важными спецификациями устройства являются значения уровней 1 и 0 тока на входе и выходе устройства, а также значения уровней 1 и 0 выходного напряжения. (С разрешения фирмы Texas Instruments.)

Электрические характеристики даны для стандартных температурных условий (если это не оговорено особо)

ПАРАМЕТР	ЗНАЧЕНИЕ	УСЛОВИЯ ИСПЫТАНИЙ ^{а)}	СЕРИЯ 54 СЕРИЯ 74		Единица измерения
			'367A'366A '367A'366A	'LS366A', 'LS366A' 'LS367A', 'LS367A'	
V_{IH} Вх. напряжен. высокого уровня	1,2	54 семейство 74 семейство	2	2	В
V_{IL} Вх. напряжен. низкого уровня	1,2		0,8	0,7	В
V_{IK} Вх. граничное напряжение	3	$V_{CC} = V_{CC \text{ мин.}}$ $V_{IH} = 2В$ $I_{IH} = I_{OH \text{ макс.}}$	-1,5	-1,5	В
V_{OH} Выходное напряжение высокого уровня	1	$V_{CC} = V_{CC \text{ макс.}}$ $V_{IH} = V_{IL \text{ макс.}}$ $I_{OH} = I_{OL \text{ макс.}}$	2,4 3,3 2,4 3,1	2,4 3,3 2,4 3,1	В
V_{OL} Выходное напряжение низкого уровня	2	$V_{CC} = V_{CC \text{ мин.}}$ $V_{IH} = 2В$ $I_{OH} = I_{OL \text{ макс.}}$ $I_{OL} = 2мА$	0,4 0,4 0,25 0,4 0,25 0,4	0,4 0,4 0,25 0,4 0,25 0,4	В
I_{OZ} Ток на выходе при очень высоком сопротивлении	19	$V_{CC} = V_{CC \text{ макс.}}$ $V_{IH} = V_{IL \text{ макс.}}$ $V_{IL} = V_{IL \text{ макс.}}$	40 -40	40 -40	мкА
I_I Ток при максимальном входном напряжении	4	$V_{CC} = V_{CC \text{ макс.}}$	1	1	мА
I_{IH} Высокий уровень входного тока	4	$V_{CC} = V_{CC \text{ макс.}}$ $V_{IH} = 2,7В$	40 20	40 20	мкА
I_{IL} Низкий уровень входного тока	4	$V_{CC} = V_{CC \text{ макс.}}$ $V_{IH} = 0,5В$	-40 -20	-40 -20	мкА
I_{OS} Ток на вых. при корот. замыкании	6	$V_{CC} = V_{CC \text{ макс.}}$ $V_{IH} = 0,4В$ $V_{IL} = 0,4В$	-1,6 -1,6	-1,6 -1,6	мА
I_{CS} Ток от источника	7	$V_{CC} = V_{CC \text{ макс.}}$	-40 -130	-40 -225	мА

- ^{а)} Для условий, определяемых как минимальное и максимальное, используйте значение, являющееся стандартным.
^{б)} Все типовые значения даны для $V_{CC} = 5В$, $T_A = 25^\circ C$.
^{в)} $I_I = -12$ мА для SN54'/SN74' и -18 мА для SN54LS'/SN74LS', SN54S/SN74S'.
^{г)} Одновременно закорачивать следует не более одного выхода; для SN54LS'/SN74LS' и SN54S'/SN74S' продолжительность закорачивания не должна превышать одной секунды.

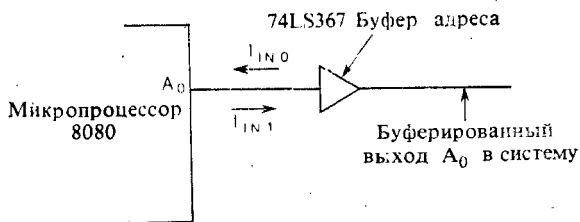


Рис. 2.8. Ток на выходе адресной линии микропроцессора должен быть согласован по величине с током на входе буфера адреса. В приведенном примере соответствующие значения тока составляют 0,4 мА для состояния логического 0 ($I_{IN\ 0}$) и 20 мкА для состояния логической 1 ($I_{IN\ 1}$). Направление тока показано стрелками.

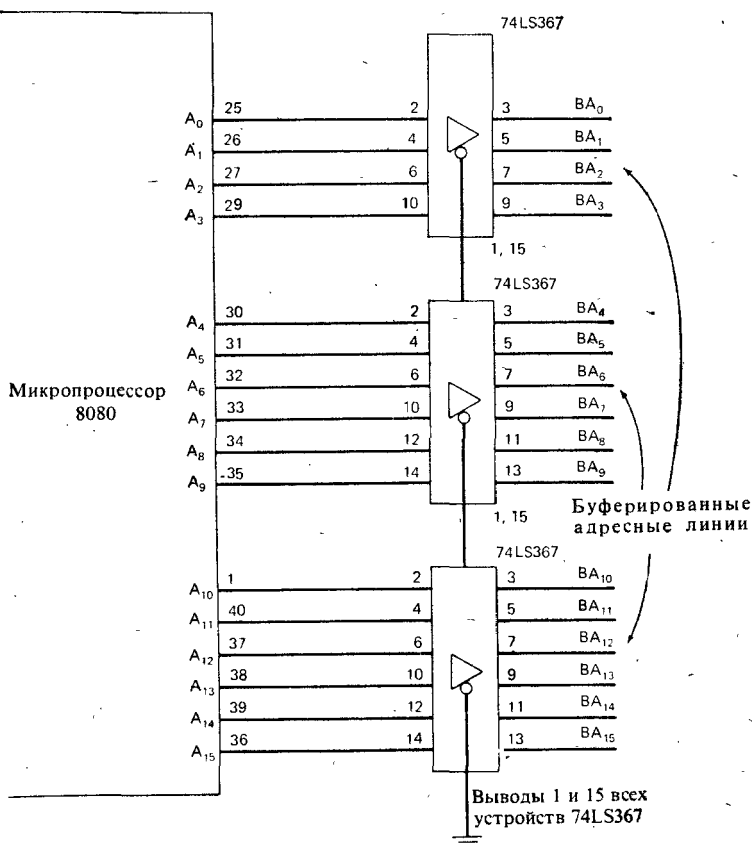


Рис. 2.9. Полная схема адресной шины микропроцессора 8080 с устройством 74LS367, используемым в качестве буфера адресных линий.

Характеристики ЦП Z80 по постоянному току
 $T_A = 0-70^\circ\text{C}$, $V_{CC} = +5 \text{ В} \pm 5\%$, если не оговорено особо

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{ILC}	Низкий уровень тактового напряжения	-0,3		0,45	В	
V_{INIC}	Высокий уровень тактового напряжения	$V_{CC}-0,6$		$V_{CC}+0,3$	В	
V_{IL}	Низкий уровень входного напряжения	-0,3		0,8	В	
V_{IH}	Высокий уровень входного напряжения	2,0		V_{CC}	В	
V_{OL}	Низкий уровень выходного напряжения			0,4	В	$I_{OL} = 1,8 \text{ мА}$
V_{OH}	Высокий уровень выходного напряжения	2,4			В	$I_{OH} = -250 \text{ мкА}$
I_{CC}	Ток от источника			150	мА	
I_{LI}	Ток утечки на входе			10	мкА	$V_{IN} = 0 \div V_{CC}$
I_{LOH}	Ток ут. на входе с тремя состояниями			10	мкА	$V_{OUT} = 2,4 \div V_{CC}$
I_{LOL}	Ток ут. на выходе с тремя состояниями			-10	мкА	$V_{OUT} = 0,4 \text{ В}$
I_{LD}	Ток ут. на шине данных в режиме ввода			± 10	мкА	$0 \leq V_{IN} \leq V_{CC}$

Характеристики ЦП Z80A по постоянному току
 $T_A = 0-70^\circ\text{C}$, $V_{CC} = +5 \text{ В} \pm 5\%$, если не оговорено особо

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{ILC}	Низкий уровень тактового напряжения	-0,3		0,45	В	
V_{INIC}	Высокий уровень тактового напряжения	$V_{CC}-0,6$		$V_{CC}+0,3$	В	
V_{IL}	Низкий уровень входного напряжения	-0,3		0,8	В	
V_{IH}	Высокий уровень входного напряжения	2,0		V_{CC}	В	
V_{OL}	Низкий уровень выходного напряжения			0,4	В	$I_{OL} = 1,8 \text{ мА}$
V_{OH}	Высокий уровень выходного напряжения	2,4			В	$I_{OH} = -250 \text{ мкА}$
I_{CC}	Ток от источника		90	200	мА	
I_{LI}	Ток утечки на входе			10	мкА	$V_{IN} = 0 \div V_{CC}$
I_{LOH}	Ток ут. на входе с тремя состояниями			10	мкА	$V_{OUT} = 2,4 \div V_{CC}$
I_{LOL}	Ток ут. на выходе с тремя состояниями			-10	мкА	$V_{OUT} = 0,4 \text{ В}$
I_{LD}	Ток ут. на шине данных в режиме ввода			± 10	мкА	$0 \leq V_{IN} \leq V_{CC}$

Рис. 2.10. Спецификация адресных выходов микропроцессора Z80 и некоторые другие спецификации по постоянному току. Напряжение на адресных выходах обозначено как V_{OL} и V_{OH} . (С разрешения фирмы Zilog.)

2.5. Адресная шина Z80

Покажем, каким образом можно построить буферы для адресной шины микропроцессора Z80. Спецификации адресных входов Z80 приведены на рис. 2.10. Из этого рисунка видно, что для адресных выходов Z80 характерно наличие тока 150 мкА в состоянии логической 1 и 2 мА в состоянии логического 0. Допустим, что адресная шина системы характеризуется нагрузкой 200 мкА в состоянии 1 и 3 мА — в состоянии 0. Как видно, в этом случае микропроцессор Z80 не в состоянии управлять адресной шиной. Заметим, снова, что подобные условия выбраны с целью сделать необходимым введение буфера адреса.

Выберем в качестве буферов адреса устройства 74LS367. Полная схема соединений для буферизированной адресной шины микропроцессора Z80 приведена на рис. 2.11. Ввиду простоты

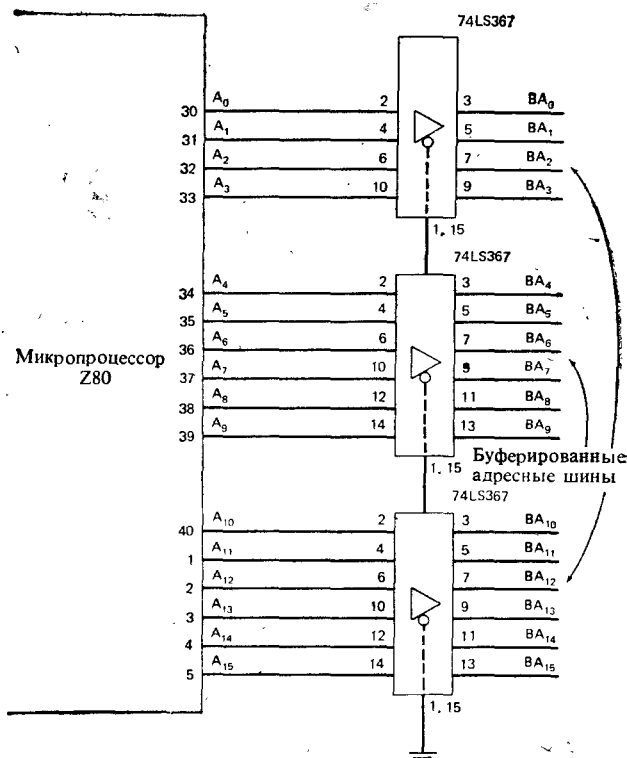


Рис. 2.11. Полная схема адресной шины микропроцессора Z80 с устройством 74LS367, используемым в качестве буфера адреса.

интерфейса Z80 с микропроцессорной системой дальнейшее рассмотрение адресной шины Z80 нецелесообразно. Отметим, что адресная шина Z80, как и 8080, состоит из 16 линий (рис. 2.9).

2.6. Адресная шина 6800

Спецификации для выходных адресных линий микропроцессора 6800 приведены на рис. 2.12, из которого видно, что нагрузка для этих линий составляет 2 мА для состояния логической 1 и 150 мкА для состояния логического 0. Допустим, что нагрузка по току в адресной шине проектируемой системы превышает возможности микропроцессора 6800, т. е. необходимо буферирование адресных выходов микропроцессора 6800. В качестве буферов адреса снова используем устройство 74LS367.

Полная схема буферированной адресной шины микропроцессора 6800 приведена на рис. 2.13. Из этой схемы видно, что адресная шина 6800, подобно шинам 8080 и Z80, состоит из 16 линий.

2.7. Адресная шина микропроцессора 8085

Адресная шина микропроцессора 8085 по своему построению отличается от уже рассмотренных шин. Для адресных шин устройств Z80, 8080, 6800 характерно то, что каждой линии шины соответствует специальный вывод в корпусе микропроцессора. Это означает, что вне зависимости от состояния микропроцессора вывод A₀ всегда соответствует разряду адресной шины A₀, что упрощает формирование 16-разрядного адреса. Здесь просто группируются 16 выводов микропроцессора, и эта группа определяется как адресная шина системы.

В устройстве 8085 используется другой принцип, основанный на «временном мультиплексировании» функций выводов, когда одни и те же выводы в разные моменты времени представляют разные функции. Это позволяет реализовать ряд дополнительных функций при тех же 40 выводах в корпусе микропроцессора.

Восемь мультиплексированных выводов устройства 8085 играют роль шины данных либо младших разрядов адресной шины. В некоторые моменты эти восемь физических выводов передают адрес, в другие моменты они используются для ввода или вывода данных. Как показано на рис. 2.14 выходы в микропроцессоре 8085 обозначаются как AD₀—AD₇.

Из рис. 2.14 видно, что старшие восемь разрядов адресной шины системы A₈—A₁₅ соответствуют аналогичным адресным выводам предыдущих микропроцессоров, т. е. каждый вывод предназначен лишь для реализации соответствующего разряда адреса.

МС 6800

Электрические характеристики ($V_{CC} = 5,0 \pm 5\%$, $V_{SS} = 0$, $T_A = 0 + 70^\circ\text{C}$, если не оговорено особо)

Характеристики	Обозначения	Мин.	Среднее	Макс.	Един. измерения
Высокий уровень входного напряжения Логика $\phi 1, \phi 2$	V_{IH} V_{IHC}	$V_{SS} + 2,0$ $V_{CC} - 0,3$	—	V_{CC} $V_{CC} + 0,1$	В пост. ток
Низкий уровень входного напряжения Логика $\phi 1, \phi 2$	V_{IL} V_{ILC}	$V_{SS} - 0,3$ $V_{SS} - 0,1$	—	$V_{SS} + 0,8$ $V_{SS} + 0,3$	В пост. ток
Положит./отрицат. выброс тактового импульса — Высокий уровень сигнала — Низкий уровень на входе	V_{OS}	$V_{CC} - 0,5$ $V_{SS} - 0,5$	—	$V_{CC} + 0,5$ $V_{SS} + 0,5$	В пост. ток
Ток утечки на входе ($V_{in} = 0 + 5,25 \text{ В}$, $V_{CC} = V_{CC \text{ макс.}}$) Логика ^{а)} ($V_{in} = 0 + 5,25 \text{ В}$, $V_{CC} = 0,0 \text{ В}$) $\phi 1, \phi 2$	I_{in}	—	1,0	100	мкА пост. ток
Ток на входе в отключенном состоянии D0—D7 ($V_{in} = 0,4 + 2,4 \text{ В}$, $V_{CC} = V_{CC \text{ макс.}}$) A0—A15, R/W	I_{ISI}	—	2,0	10 100	мкА пост. ток
Высокий уровень выходного напряжения ($I_L = 205 \text{ мкА}$, $V_{CC} = V_{CC \text{ мин.}}$) D0—D7 ($I_L = -145 \text{ мкА}$, $V_{CC} = V_{CC \text{ мин.}}$) A0—A15 R/W, VMA ($I_L = -100 \text{ мкА}$, $V_{CC} = V_{CC \text{ мин.}}$) BA	V_{OH}	$V_{SS} + 2,4$ $V_{SS} + 2,4$ $V_{SS} + 2,4$	— — —	— — —	В пост. ток
Низкий уровень выходного напряжения ($I_L = 1,6 \text{ мА}$, $V_{CC} = V_{CC \text{ мин.}}$)	V_{OL}	—	—	$V_{SS} + 0,4$	В пост. ток
Потери мощности	P_D	—	0,600	1,2	
Емкость ^{б)} ($V_{in} = 0$, $T_A = 25^\circ\text{C}$, $f = 1 \text{ МГц}$) $\phi 1, \phi 2$ TSC DBE D0—D7 Логические входы A0—A15, R/W, VMA	C_{in} C_{out}	80 — — — —	120 — 7,0 10 6,5	160 15 10 12,5 12	пкФ пкФ
Рабочая частота	f	0,1	—	1,0	МГц
Тактирование					
Время цикла	t_{cyc}	1,0	—	10	мкс
Длительность тактового импульса (Измерено при $V_{CC} = -0,3 \text{ В}$) $\phi 1$ $\phi 2$	$PW_{\phi H}$	430 450	— —	4500 4500	нс
Общее для $\phi 1$ и $\phi 2$	t_{ut}	940	—	—	нс
Время нарастания и спада $\phi 1, \phi 2$ (Измерено между $V_{SS} + 0,3 \text{ В}$ и $V_{CC} - 0,3 \text{ В}$)	$t_{\phi 1, \phi 2}$	5,0	—	50	нс
Время задержки или разделение по частоте (Измерено при $V_{OV} = V_{SS} + 0,5 \text{ В}$)	t_d	0	—	9100	нс
Продолжительность выброса	t_{os}	0	—	40	нс

а) За исключением $\overline{IRQ}$ и $\overline{NM\overline{I}}$, требующих согласующих резисторов сопротивлением 3кОм для обеспечения нагрузочной способности точечного ИЛИ-соединения в оптимальном режиме

б) Емкость периодически замеряется

Рис. 2.12. Спецификации адресных выходов микропроцессора 6800. На адресных выходах дано напряжение высокого и низкого уровня сигнала. (С разрешения фирмы Motorola.)

Ввиду того что в микропроцессоре 8085 используются мультиплексированные выходы для адреса и данных, формирование младших восьми разрядов адресной шины по сравнению с ра-

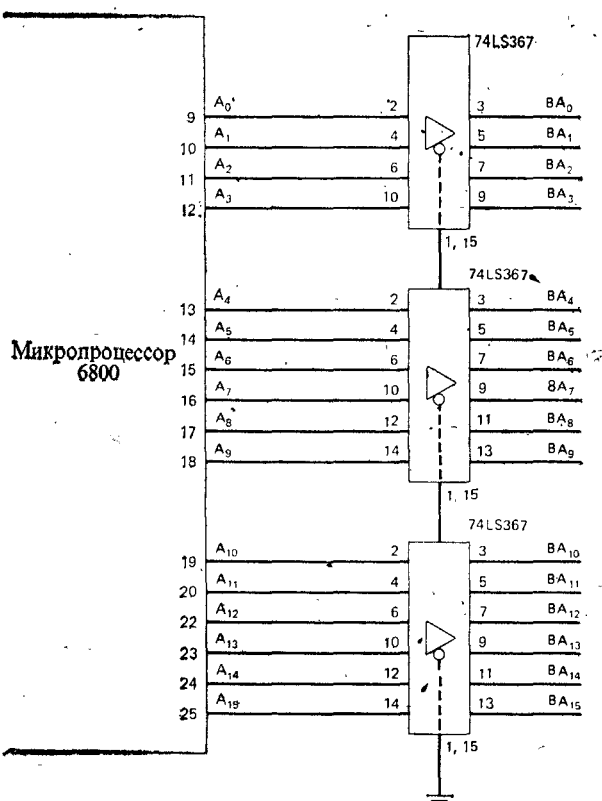


Рис. 2.13. Полная схема адресной шины микропроцессора 6800 с устройством 74LS367, используемым в качестве буфера адреса.

нее рассмотренными микропроцессорами несколько усложняется. Главная особенность заключается в необходимости «фиксации» логического состояния выводов AD₀—AD₇ микропроцессора 8085 в моменты, когда они функционально представляют адресные разряды A₀—A₇. Для этого необходимо точно знать, когда на этих выводах отображается адресная информация.

В корпусе 8085 существует специальный вывод (№ 30), обозначаемый ALE (Address Latch Enable) — открытие фиксатора адреса, сигнал на котором в нормальном состоянии соответст-

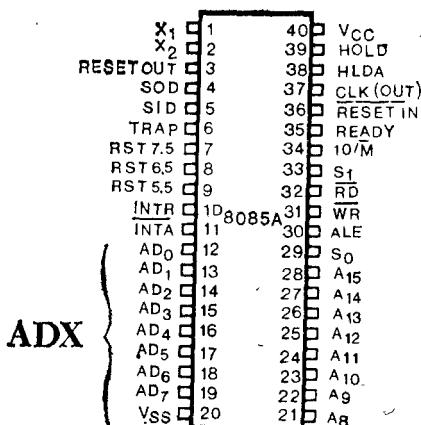


Рис. 2.14. Обозначения выводов для микропроцессора 8085. Отметим наличие выводов, обозначаемых AD_0 — AD_7 . Эти выводы относятся к шине данных с временным мультиплексированием, а также служат для передачи младшего байта адреса. С разрешения фирмы Intel.)

ует логическому 0. Если информация на выводах 8085 AD_0 — AD_7 является адресной — A_0 — A_7 , то ALE переводится в состояние 1. При переходе ALE из состояния 1 в состояния 0 информация на AD_0 — AD_7 должна быть зафиксирована. Это видно из временной диаграммы рис. 2.15.

На рис. 2.16 показано, каким путем осуществляется фиксация данных техническими средствами применительно к полной информации 16-разрядной адресной шины для микропроцессора 8085. Отметим, что необходимость в использовании буферов с тремя состояниями 74LS367 возникает лишь для старших разрядов адреса A_8 — A_{15} , поскольку младшие разряды буферизуются с помощью фиксатора 74LS374. Устройство 74LS374, выполненное в виде одной интегральной схемы, является одновременно фиксатором и буфером.

Сигнал ALE на рис. 2.16 инвертируется за счет использования фиксатора с импульсным запуском. В соответствии со спецификациями 8085 адресные данные должны фиксироваться по заднему фронту сигнала ALE. С другой стороны, согласно спецификациям 8-разрядного фиксатора 74LS374, это устройство фиксирует данные по переднему фронту тактового импульса. Поэтому необходимо инвертировать сигнал ALE, поступающий от микропроцессора. Это позволит осуществить физически фиксирование информации AD_0 — AD_7 в 8-разрядных фиксаторах 74LS374 по заднему фронту сигнала ALE.

Фиксаторы другого типа, иногда используемые для выполнения этой функции, позволяют пропускать данные на выходы Q при состоянии тактового импульса, соответствующем логической 1. При состоянии же 0 тактового импульса данные на выходе устройства сохраняются. Примером такого устройства является фиксатор 74LS373, пропускающий адресную информацию A_0 — A_7 по мере того, как она генерируется микропроцессо-

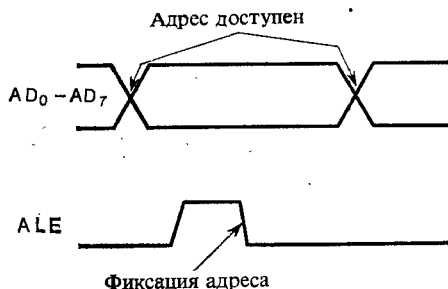


Рис. 2.15. Временная диаграмма, определяющая взаимосвязь между сигналом ALE и адресной информацией на выводах AD₀—AD₇ микропроцессора 8085.

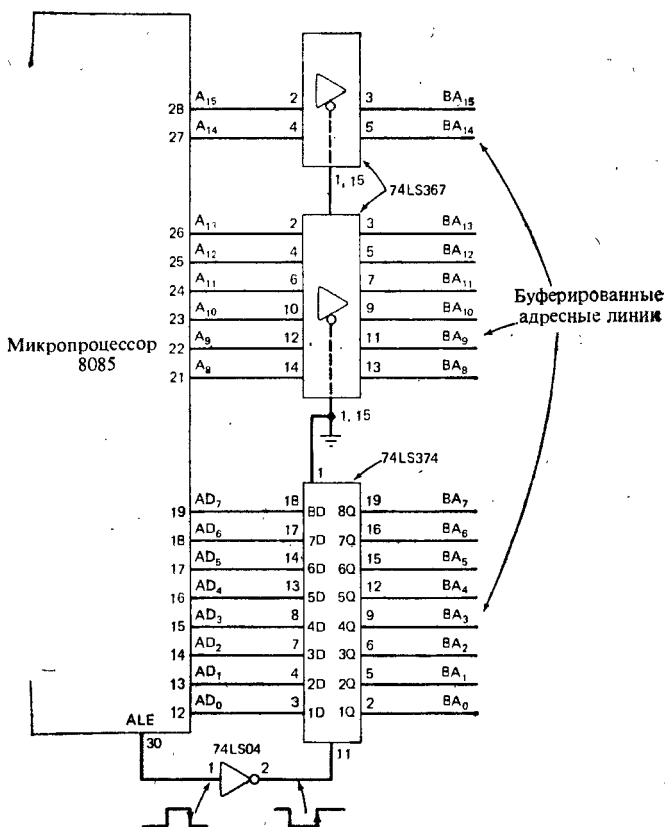


Рис. 2.16. Полная схема адресной шины микропроцессора 8085. При выдаче младшего байта адреса используется восьмиразрядный фиксатор 74LS374.

ром 8085. Различия во временных соотношениях для фиксаторов 74LS374 и 74LS373 показаны на рис. 2.17.

Отметим, что для стробирования адресной информации от микропроцессора 8085 может быть использован любой фиксатор. Единственная предосторожность, которую необходимо со-

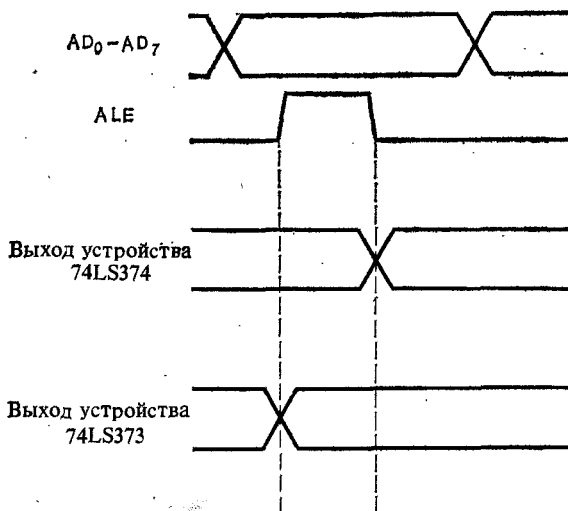


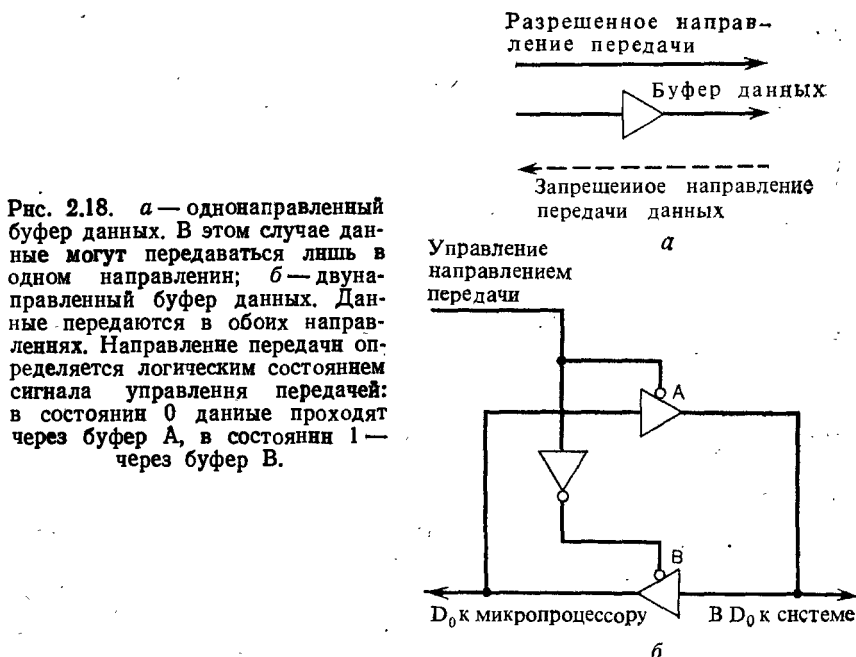
Рис. 2.17. Временная диаграмма, определяющая взаимосвязь сигналов при выдаче адреса, когда вместо устройства 74LS374 используется фиксатор 74LS373.

блюдать при использовании фиксаторов разных типов, заключается в согласовании нагрузки по току для входов фиксатора и выводов AD_0-AD_7 микропроцессора 8085 во избежание их перегрузки, т. е. необходимо убедиться, что ток на входе используемого фиксатора не является слишком большим для микропроцессора 8085.

2.8. Пояснения к шине данных системы

Как уже было показано, шина данных является двунаправленной. Это означает, что в некоторые промежутки времени информация вводится в микропроцессор (когда последний функционирует в режиме ввода), в другие же промежутки времени микропроцессор генерирует информацию и затем выводит ее в систему (работа в режиме вывода). Когда микропроцессор рабо-

тает в режиме вывода, шина данных часто соединяется параллельно со многими входами. При этом существует возможность того, что общая нагрузка от всех входов будет для шины данных чрезмерной. Эта ситуация подобна той, которую мы обсу-



дали при рассмотрении нагрузки на адресную шину. Для ее предотвращения вновь можно организовать буферирование.

Однако для случая шины данных необходимо рассмотреть дополнительный фактор, а именно, то, что эта шина данных в отличие от адресной шины не является однонаправленной. Она двунаправленная, и этот факт исключает возможность использования техники буферирования, применяемой в случае адресной шины. Рассмотрим рис. 2.18, а.

Из рис. 2.18, а видно, что рассмотренное выше буферирование может быть использовано лишь в том случае, когда информация по шине данных выдается в систему. Если же данные из системы будут передаваться в микропроцессор, они будут «блокированы» и не достигнут входов микропроцессора. На рис. 2.18, б показана техника буферирования для двунаправленной шины, основанная на использовании логических схем с тремя состояниями. Отметим появление дополнительного сигнала

управления, логическое состояние которого задает направление буферизированных данных. Например, при сигнале управления, соответствующем логической 1, разрешается передача данных из системы к выводам микропроцессора. В случае же, когда состояние этого сигнала соответствует логическому 0, разрешается передача данных от микропроцессора в систему через буфер А. Если один буфер с тремя состояниями открыт, другой в это время закрыт.

Схема рис. 2.18, б иллюстрирует общие принципы буферирования при двунаправленной передаче, которые могут быть использованы применительно к шине данных микропроцессора. Обсудим, каким образом описанная техника буферирования может быть реализована для четырех рассматриваемых микропроцессоров. Будем помнить, что буферирование при двунаправленной передаче требует сигнала управления для индикации направления передачи. Ниже будет показано генерирование этого сигнала для различных микропроцессоров. Таким образом, решение проблемы буферирования для двунаправленной шины показано на рис. 2.18, б.

2.9. Буферизированная шина данных микропроцессора 8080

При использовании микропроцессора 8080 буферирование шины данных не вызывает беспокойства. Этот микропроцессор предназначен для использования совместно со специальной ин-

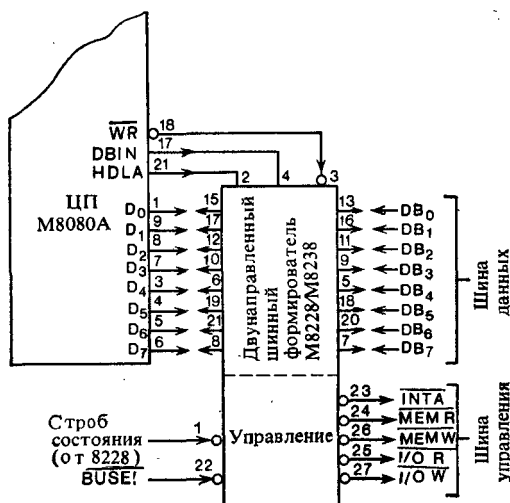


Рис. 2.19. Схема соединения системного контроллера 8228 с микропроцессором 8080. (Устройство 8228 выполняет все функции буферирования шины данных и управления направлением передачи.) (С разрешения фирмы Intel.)

тегральной схемой, называемой «системный контроллер и шинный формирователь типа 8228».

Короче говоря, устройство 8228 осуществляет буферирование шины данных и автоматический контроль направления передачи данных. Поэтому применяя микропроцессор 8080, желательно использовать и устройство 8228. На рис. 2.19 показано

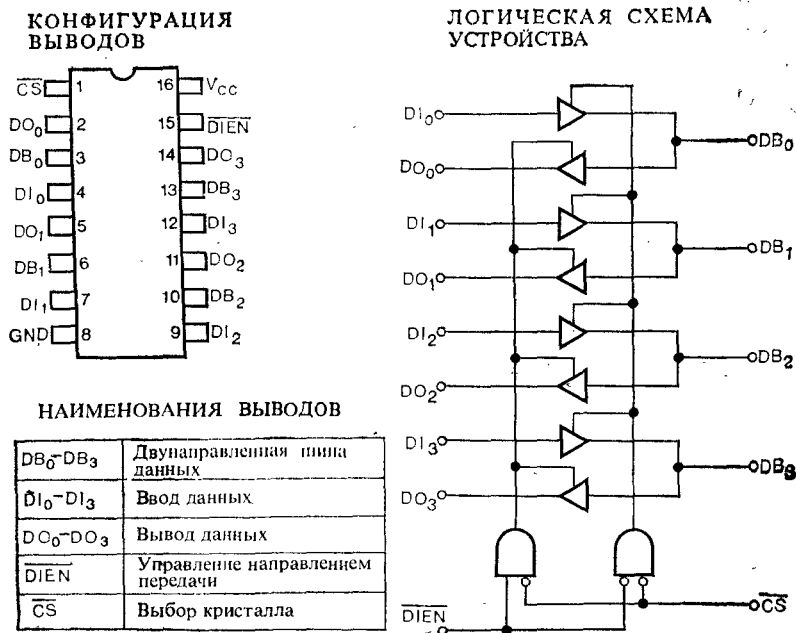


Рис. 2.20. Выводы и блок-схема устройства 8216. (С разрешения фирмы Intel.)

совместное соединение устройств 8228 и 8080 при построении шины данных системы.

Обсудим теперь, каким образом можно построить буферизованную шину данных без использования устройства 8228. Это позволит лучше понять назначение различных сигналов управления в микропроцессорной системе на базе устройства 8080. Однако при использовании микропроцессора 8080 наиболее простым решением является применение системного контроллера и шинного формирователя 8228. Поэтому настоящее обсуждение вызвано лишь стремлением глубже разобраться в особенностях управления техническими средствами системы со стороны микропроцессора 8080.

Двунаправленные буферы для шины данных оформлены в виде двух отдельных интегральных схем, именуемых устройст-

вами 8216. Выводы такого устройства показаны на рис. 2.20. Видим, что это устройство состоит из формирователей с тремя состояниями, которые похожи на схемы, показанные на рис. 2.18, б. Вывод $\overline{CS}$ (№ 1) позволяет устройству функционировать в двунаправленном режиме, когда сигнал на этом выводе соответствует логическому 0.

Допустим на время, что вывод $\overline{CS}$ всегда находится в состоянии 0. Существуют ситуации, когда необходимо отключить шину данных. Эти ситуации будут рассмотрены в последующих главах.

Отметим на рис. 2.20 вывод $\overline{DIEN}$ (№ 15), определяющий направление передачи в устройстве. Сигнал, поступающий на этот вывод, эквивалентен сигналу управления направлением передачи данных, показанному на рис. 2.18. Если этот вывод находится в состоянии логического 0, то передача данных разрешена в направлении от DI к DB. При противоположном состоянии вывода (логическая 1) разрешается передача от DB к DO. Выводы DI и DO физически соединены вместе.

Подсоединим устройство к микропроцессору 8080, как это показано на рис. 2.21. Очевидно, на вывод $\overline{DIEN}$ устройства 8216 необходимо подать некоторый управляющий сигнал, генерируемый микропроцессором. В случае микропроцессора 8080 при сигнале на выводе DBIN, соответствующем логической 1, осуществляется ввод данных. Поэтому можно подать сигнал DBIN на линию $\overline{DIEN}$ устройства 8216. Однако при этом необходимо убедиться, что выходная нагрузка для сигнала DBIN не превышена. Устройство 8216 потребляет ток не более 0,5 мА в состоянии логического 0 и не более 80 мкА в состоянии логической 1. Выходной сигнал DBIN микропроцессора 8080 характеризуется 1,9 мА в состоянии 0 и 150 мкА в состоянии 1. Отсюда ясно, что с помощью выходной линии DBIN возможно управлять входом устройства 8216. Необходимо постоянно помнить о максимальной допустимой нагрузке для каждой линии в системе в любой заданный момент времени и не допускать превышения этой величины. Это особенно важно для начинающих.

При вводе данных в микропроцессор 8080 сигнал DBIN соответствует состоянию логической 1. Это означает, что выводы DI, DO устройства 8216 должны быть соединены с шиной данных микропроцессора, что соответствует ранее приведенному объяснению процесса прохождения данных в устройстве 8216. На рис. 2.21 приведена полная схема буферирования шины данных 8080 без использования системного контроллера 8228.

Еще раз подчеркнем, что устройство 8228 отличается простотой применения, и этот фактор следует прежде всего рассматривать при использовании микропроцессора 8080. Устрой-

ство 8228 позволяет снизить общее число выводов в системе, повысить ее надежность и уменьшить вероятность возникновения неполадок.

2.10. Буферированная шина данных микропроцессора Z80

Для буферирования шины данных микропроцессора Z80 будем использовать технику, подобную той, которая применялась в случае микропроцессора 8080. Для Z80 отсутствует специальный системный контроллер, подобный устройству 8228. Для реализации соответствующего буферирования Z80 используем устройство 74LS245. Для нашего рассмотрения это устройство открывается путем подачи на вывод 19 сигнала уровня логического 0. На вывод DIR, определяющий направление передачи (вывод 1) устройства 74LS245, подадим управляющий сигнал $\overline{RD}$ с вывода 21 микропроцессора Z80. Все это показано на рис. 2.22.

Состояние логического 0 на $\overline{RD}$ определяет нахождение микропроцессора в режиме приема данных. В этом случае устройство 74LS245 должно разрешать передачу данных от системы на шину данных Z80. Вспомним, что в системе 8080 для управления направлением передачи данных использовался сигнал DBIN (активизирован при логической 1). Таким образом, для сигнала на прием данных от микропроцессора Z80 активное состояние соответствует логическому 0, а от микропроцессора 8080 — логической 1.

2.11. Буферированная шина данных микропроцессора 6800

Для буферирования шины данных микропроцессора 6800 снова используем устройство 74LS245. (Отметим, что для этой цели можно применить любую технику буферирования, приемлемую при двунаправленной передаче.) Использование 74LS245 мотивируется стремлением уменьшить число выводов в системе, но не следует думать, что это единственное устройство, подходящее для этой цели.

В рассматриваемом случае на вывод 19, открывающий устройство 74LS245, подается сигнал, соответствующий логическому 0, либо этот вывод заземляется. При этом 74LS245 остается открытым.

Линия управления направлением передачи устройства 74LS245 соединяется с выводом R/W № 34 микропроцессора 6800, как это показано на рис. 2.23. Вывод R/W находится в состоянии логической 1, если режим микропроцессора соответствует вводу данных из системы. Отметим, что все рассматри-

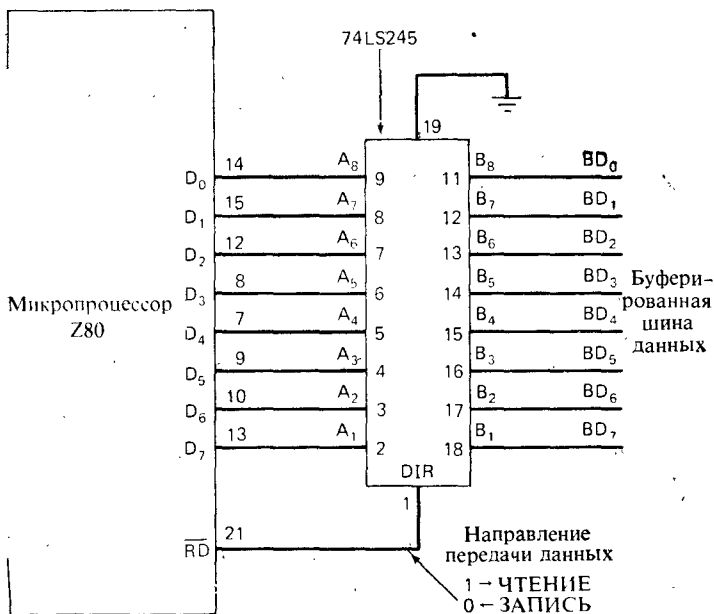


Рис. 2.22. Полная схема буферирования шины данных микропроцессора Z80 с помощью устройства 74LS245.

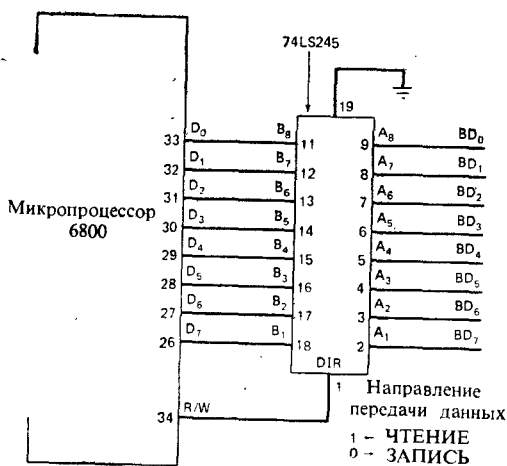


Рис. 2.23. Полная схема буферирования шины данных микропроцессора 6800. Отметим аналогию с рис. 2.22 в использовании устройства 74LS245.

Вспомним, что по шине управления системы передаются четыре сигнала, а именно:

1. ЧТЕНИЕ ИЗ ПАМЯТИ.
2. ЗАПИСЬ В ПАМЯТЬ.
3. ЧТЕНИЕ С УСТРОЙСТВА ВВОДА.
4. ЗАПИСЬ В УСТРОЙСТВО ВЫВОДА.

Покажем теперь, каким образом реализуются эти сигналы для каждого микропроцессора.

2.14. Шина управления системы на базе микропроцессора 8080

При рассмотрении шины данных микропроцессора 8080 было обнаружено, что буферирование шины данных и необходимое управление направлением передачи обеспечивались специаль-

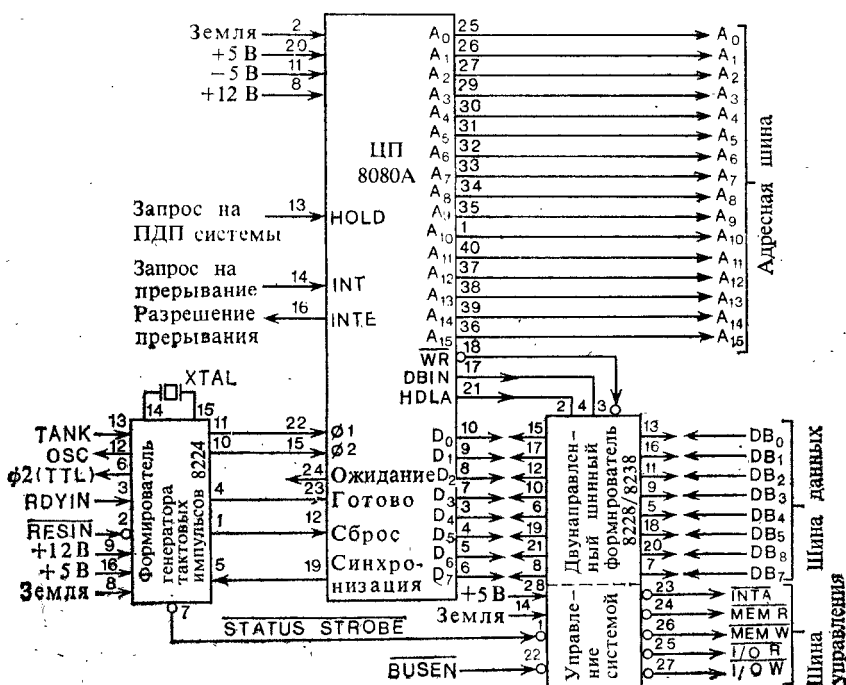


Рис. 2.25. Схема формирования шины управления на выходе системного контроллера 8228. (С разрешения фирмы Intel.)

ным устройством — системным контроллером и шинным формирователем 8228. Это же устройство генерирует в нужные моменты времени четыре основных сигнала управления, приведенных

выше, что показано на рис. 2.25, где упомянутые сигналы управления обозначаются как $\overline{\text{MEMW}}$, $\overline{\text{MEMR}}$, $\overline{\text{I/OR}}$, $\overline{\text{I/OW}}$. Эти сигналы являются взаимно исключающими, а их активное состояние соответствует уровню логического 0. Таким образом, никакие два из них не могут быть одновременно в состоянии 0.

При использовании микропроцессора 8080 в качестве контроллера системы устройство 8228 позволяет значительно упростить проектирование системы. Однако с целью оценки различных альтернативных вариантов рассмотрим, каким образом четыре сигнала управления могут быть сгенерированы с помощью дискретных цифровых устройств. Это поможет лучше понять, как микропроцессор 8080 управляет техническими средствами системы.

2.15. Фиксатор состояния микропроцессора 8080

Центральный процессор 8080 генерирует на выводы шины данных байт информации, называемый словом состояния. Слово состояния генерируется для того, чтобы оповестить технические средства системы о намерениях микропроцессора в течение первого T-цикла M-цикла¹⁾. При переходе микропроцессора 8080 в состояние чтения данных из памяти вначале для индикации этого состояния генерируется слово состояния. Это слово используется системой для установки правильного схемного соединения, используемого микропроцессором.

При выдаче слова состояния на шину данных микропроцессор 8080 информирует об этом систему путем перевода в состояние логической 1 сигнала синхронизации на выводе № 19. Этот сигнал используется системой для формирования строба, пропускающего данные с шины данных в фиксатор, называемый фиксатором состояния, что показано на рис. 2.26.

После фиксации слова состояния определенные биты информации состояния логически объединяются с сигналами $\overline{\text{DBIN}}$ и $\overline{\text{WR}}$ с целью формирования четырех сигналов шины управления. Схематически это показано на рис. 2.27.

Теперь шина управления системы в случае микропроцессора 8080 реализована с помощью схем цифровой логики. Необходимо подчеркнуть, что шина управления должна обеспечивать управление всех соединенных с нею входов. Поэтому необходимо буферирование шины управления. Для этого можно использовать контроллер 8228. Если шина управления построена с помощью цифровых логических схем, как на рис. 2.27, то для ее буферирования можно использовать стандартное устройство 7400.

¹⁾ Подробнее см. Coffron J. W., Understanding and Troubleshooting the Microprocessor, Prentice-Hall, Englewood Cliffs, N. J., 1980.

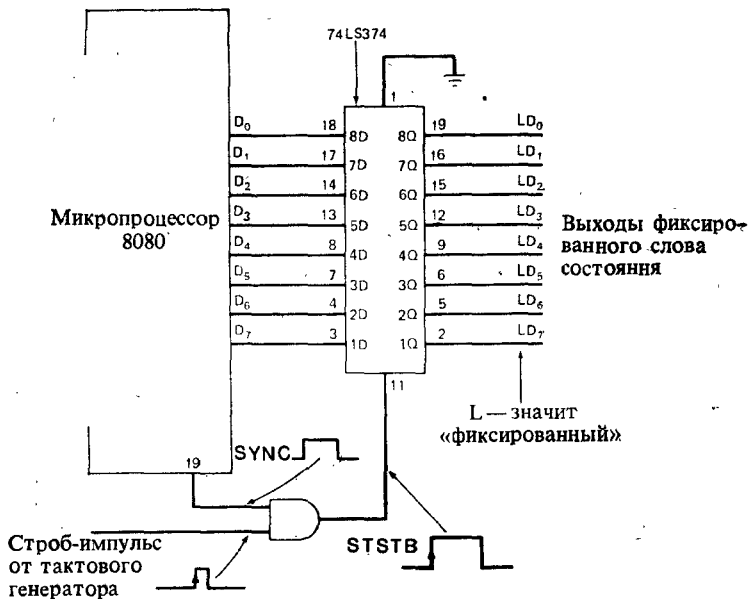


Рис. 2.26. Схема фиксации слова состояния с помощью схем дискретной логики. Фиксация слова состояния устройством 74LS374 осуществляется по уровню 1 синхронизирующего импульса.

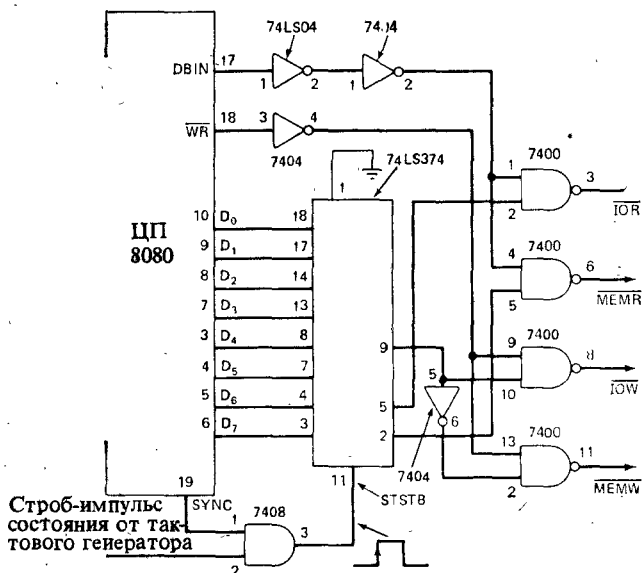


Рис. 2.27. Схема реализации шины управления системы на базе микропроцессора 8080 с помощью схем дискретной логики.

Таблица 2.2

Логические условия для выходов управляющих сигналов микропроцессора Z80 при указанных функциях системы

№ 19	№ 20	№ 21	№ 22		
$\overline{MREQ}$	$\overline{IORQ}$	$\overline{RD}$	$\overline{WR}$		
1	0	1	1	IOR	} Сокращенное обозначение функций системы
1	0	0	0	IOW	
0	1	1	1	MEMR	
0	1	0	0	MEMW	

2.16. Шина управления системы на базе Z80

Для управления в системе на базе микропроцессора Z80 используются сигналы, обозначаемые $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$. Логическая комбинация этих четырех сигналов позволяет реали-

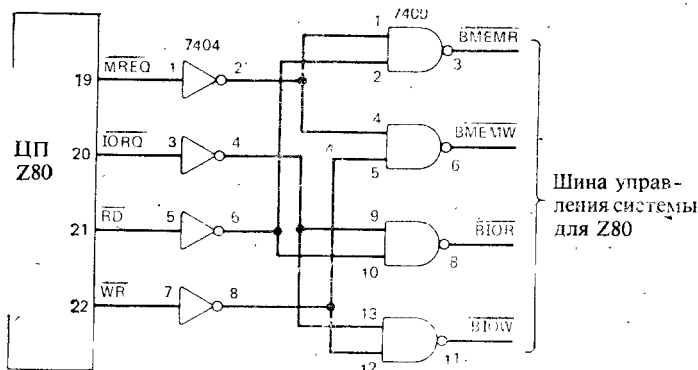


Рис. 2.28. Полная схема реализации шины управления системы на базе микропроцессора Z80 с помощью схем дискретной логики.

зовать шину управления системы. Возможные логические комбинации указанных сигналов приведены в табл. 2.2. Примерная реализация этих четырех сигналов управления показана схематически на рис. 2.28.

2.17. Шина управления системы на базе 8085

Для построения шины управления микропроцессора 8085 используем управляющие выходы, обозначаемые как $\overline{IO/\overline{M}}$ (вывод 34), $\overline{RD}$ (вывод 32) и $\overline{WR}$ (вывод 31). В табл. 2.3 приведены логические состояния сигналов на этих выводах соответственно функциям шины управления.

Таблица 2.3

Логические условия для выводов управляющих сигналов микропроцессора 8085 при указанных функциях системы

№ 32	№ 31	№ 34		
$\overline{RD}$	$\overline{WR}$	$IO/\overline{M}$		
0	1	1	IOR	Сокращенное обозначение функций системы
1	0	1	IOW	
0	1	0	MEMR	
1	0	0	MEMW	

Из табл. 2.3 видно, что при выполнении операций чтения или записи для устройства ввода-вывода (IOR и IOW) сигнал на выводе $IO/\overline{M}$ (34) соответствует состоянию логической 1.

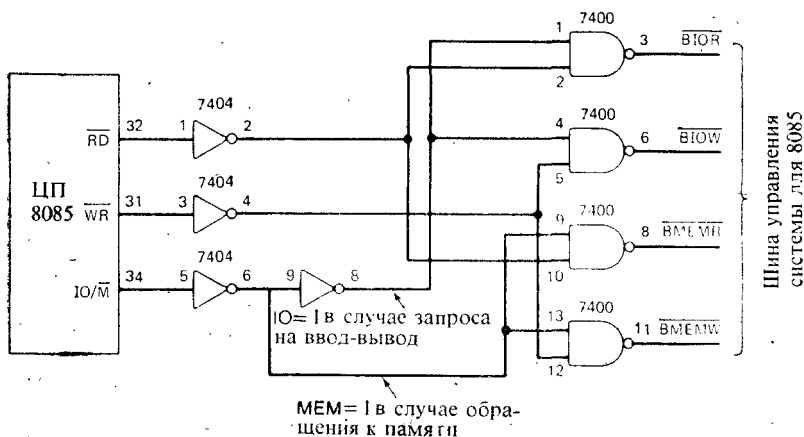


Рис. 2.29. Полная схема реализации шины управления системы на базе микропроцессора 8085 с помощью схем дискретной логики.

При выполнении же операций чтения и записи в память ($\overline{MEMR}$ и $\overline{MEMW}$) этот сигнал соответствует состоянию 0.

При реализации функций, соответствующих обозначениям этих выводов (чтение и запись), на выходы $\overline{RD}$ и $\overline{WR}$ (32 и 31) подается сигнал 0. Таким образом, при выполнении чтения информации из памяти или с устройства ввода сигнал $\overline{RD}$ соответствует логическому 0. При выполнении же записи в память или на устройство вывода сигнал $\overline{WR}$ соответствует логическому 0. Логические схемы, необходимые для реализации управляющих сигналов в системе на базе микропроцессора 8085, приведены на рис. 2.29.

2.18. Шина управления системы на базе 6800

Микропроцессоры 8080, 8085 и Z80 спроектированы таким образом, что выполнение функций доступа к памяти и к устройствам ввода-вывода осуществляется отдельно. Эти микропроцессоры имеют специальные команды, реализующие электрическую связь с памятью либо с устройствами ввода-вывода системы, что позволяет достаточно просто построить шину управления системы. Было принято, что во всех трех микропроцессорах используется стандартная архитектура реализации операций ввода-вывода.

В микропроцессоре 6800 не существует специальных команд, устанавливающих электрическую связь либо с памятью, либо с устройствами ввода-вывода. Этот микропроцессор не различает память и устройства ввода-вывода; т. е. не различает тип схем, осуществляющих прием или генерирование данных. Все передачи в системе обрабатываются как обращения к памяти. Архитектура ввода-вывода подобного рода известна как ввод-вывод по аналогии с обращением к памяти. Для рассмотрения этой архитектуры введем специальный раздел с описанием адресного пространства памяти, используемого устройствами ввода-вывода.

Адресное пространство микропроцессора 6800 включает 64К (2¹⁶) адресов. Отведем часть из этих 64К адресов для использования устройствами ввода-вывода. При конкретном распределении ячеек памяти полезно пользоваться так называемой таблицей распределения памяти. С приобретением достаточного опыта в проектировании микропроцессорных систем распределение памяти осуществляется достаточно просто. Тем не менее из соображений методического характера для микропроцессора 6800 вопросы распределения памяти целесообразно рассмотреть детально.

2.19. Распределение памяти

Распределением памяти называют разбиение имеющегося объема памяти в системе с целью реализации отдельных функций. Эта процедура характерна для любого микропроцессора. На распределение памяти влияет целый ряд факторов, некоторые из которых перечислены ниже: принятая архитектура ввода-вывода (линейная выборка, ввод-вывод по аналогии с обращением к памяти, стандартный ввод-вывод); используемые в системе типы постоянных и оперативных запоминающих устройств (ПЗУ и ОЗУ); частные особенности функционирования микропроцессора. Подробно распределение памяти в системе будет рассмотрено на примере проектирования небольшой системы, управляемой микропроцессором.

Прежде всего обсудим некоторые особенности микропроцессора 6800 с целью обоснования конкретного распределения памяти. Вначале необходимо рассмотреть, каким образом микропроцессор начинает работу. С подачей питания микропроцессор переводится в рабочий режим с передачей управления в заранее определенную ячейку памяти, соответствующую адресу FFFE. Микропроцессор читает данные из ячеек FFFE и FFFF и формирует команду перехода по адресу, записанному в эти ячейки. После этого выполняется команда, расположенная по адресу перехода.

Предположим для примера, что в ячейке FFFE записано число 08, а в ячейке FFFF — 00. После считывания этих чисел управление передается в ячейку 0800 и начинается выполнение записанной там команды. Поскольку в описанной процедуре используются большие адреса, то область памяти, выделяемая для ввода-вывода, не должна располагаться в верхней части памяти.

Отведем область памяти с адресами от 0400 до 07FF для использования в качестве адресов ввода-вывода. При этом возникает необходимость в распознавании этого адресного пространства и генерировании соответствующего сигнала (IO/M). Для формирования сигналов управления этот сигнал комбинируется с сигналом R/W микропроцессора. Описанная процедура показана на схеме рис. 2.30.

На рис. 2.31 приведена полная схема реализации шины управления системы на базе микропроцессора 6800. Выбор адресного пространства 0400—07FF в качестве адресов ввода-вывода не является обязательным, а осуществлен лишь в иллюстративных целях.

На рис. 2.31 показан также сигнал VMA, используемый для формирования шины управления системы. Этот сигнал генерируется микропроцессором и называется сигналом достоверности адреса памяти (Valid Memory Address). При выполнении команд на адресные линии может быть выдан несуществующий адрес. Если же сигнал VMA соответствует логической 1, то адрес, выданный микропроцессором, соответствует реально существующей ячейке ПЗУ или ОЗУ. Поэтому во избежание ложной записи в память системы необходимо использовать генерируемый микропроцессором сигнал VMA, уточняющий все сигналы на шине управления.

Отметим также использование при функционировании шины управления тактового сигнала фазы 2 (ϕ_2). Из временных диаграмм микропроцессора 6800 можно заключить, что процессы в системе возбуждаются по заднему фронту импульса ϕ_2 . Тем самым передача данных от микропроцессора в систему и наоборот начинается по заднему фронту импульса ϕ_2 . Поэтому импульс ϕ_2 используется для формирования сигналов шины уп-

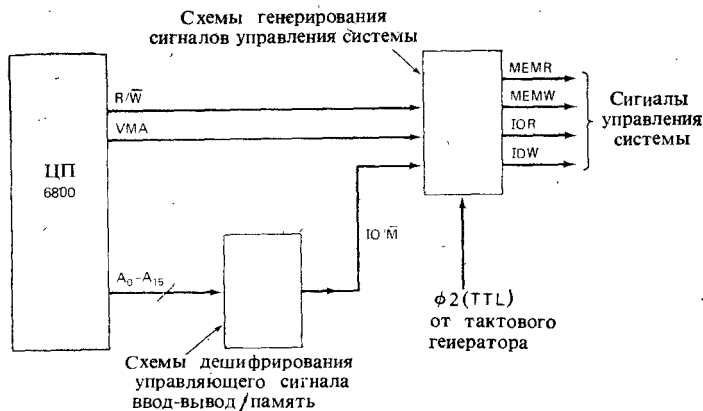


Рис. 2.30. Схема генерирования с помощью схем дешифрирования сигнала $IO/\bar{M}$ в системе на базе микропроцессора 6800. Наличие этой схемы обусловлено архитектурой ввода-вывода 6800 по аналогии с обращением к памяти.

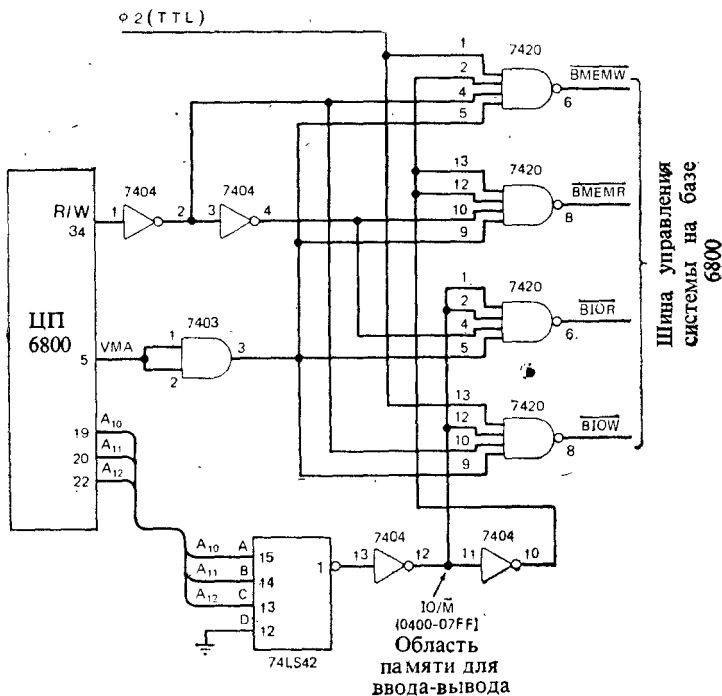


Рис. 2.31. Полная схема реализации шины управления системы на базе микропроцессора 6800 с помощью схем дискретной логики, построенная на основе блок-схемы рис. 2.30.

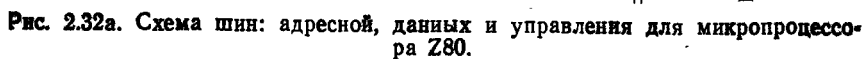


Рис. 2.32а. Схема шин: адресной, данных и управления для микропроцессора Z80.

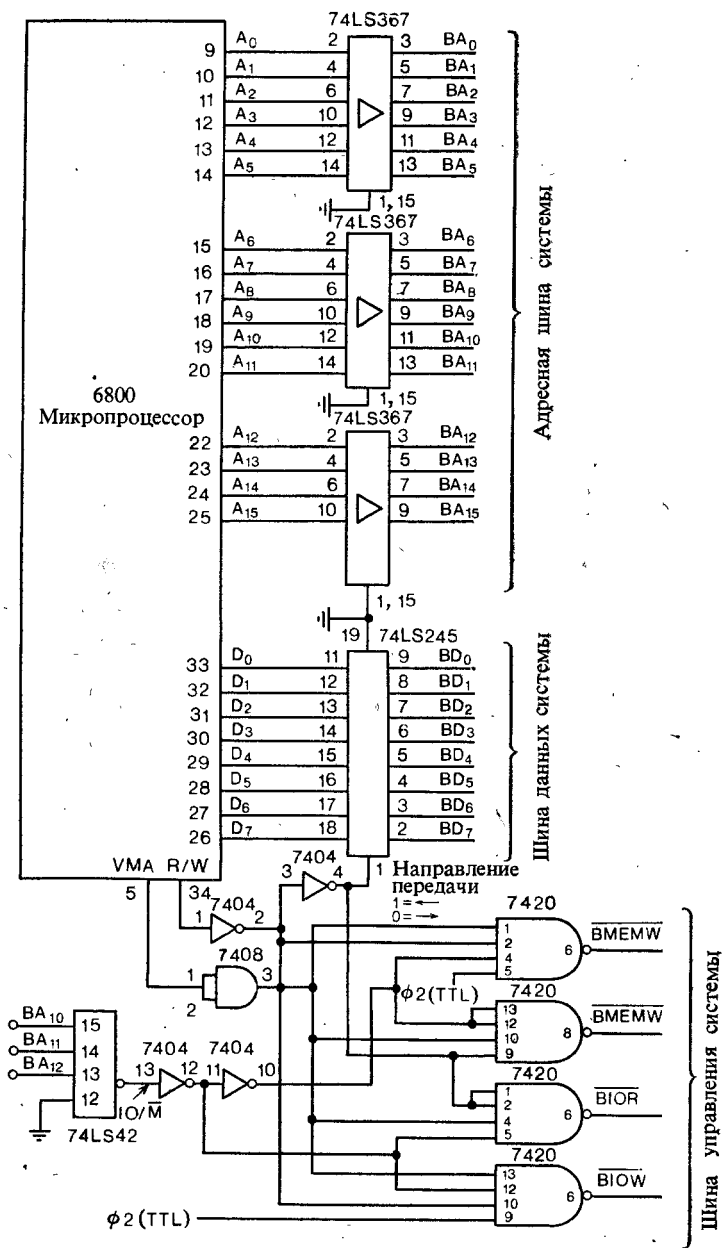


Рис. 2.326. Схема шин: адресной, данных и управления для микропроцессора 6800.

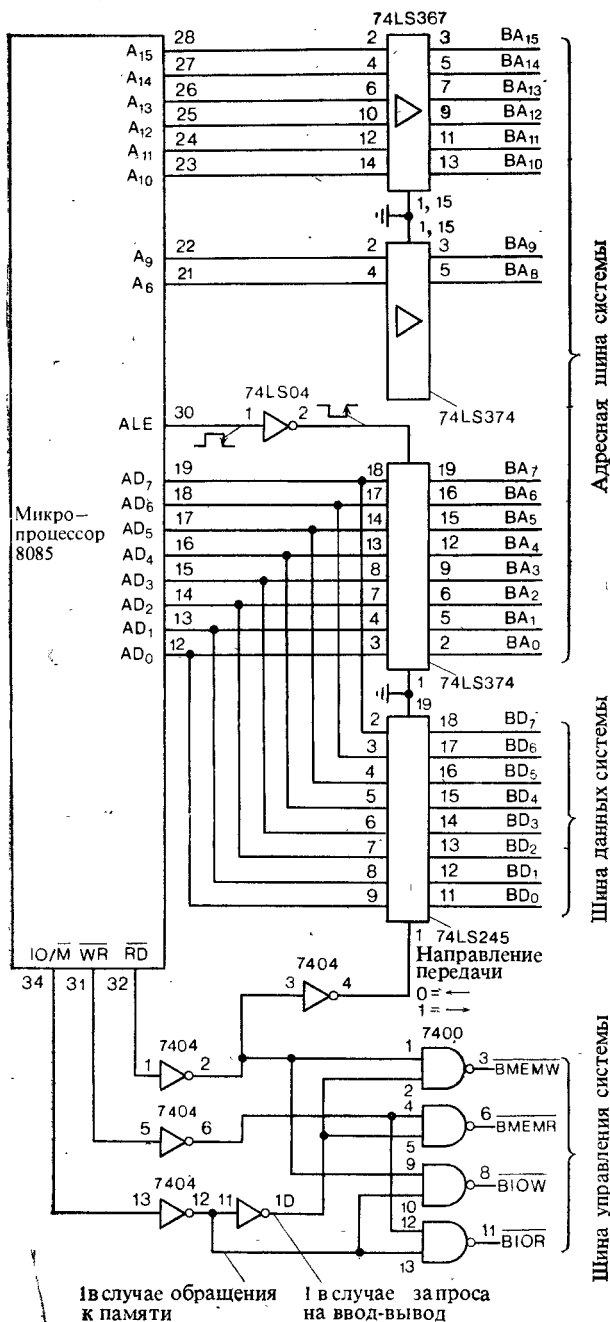


Рис. 2.32в. Схема шин: адресной, данных и управления для микропроцессора 8085.

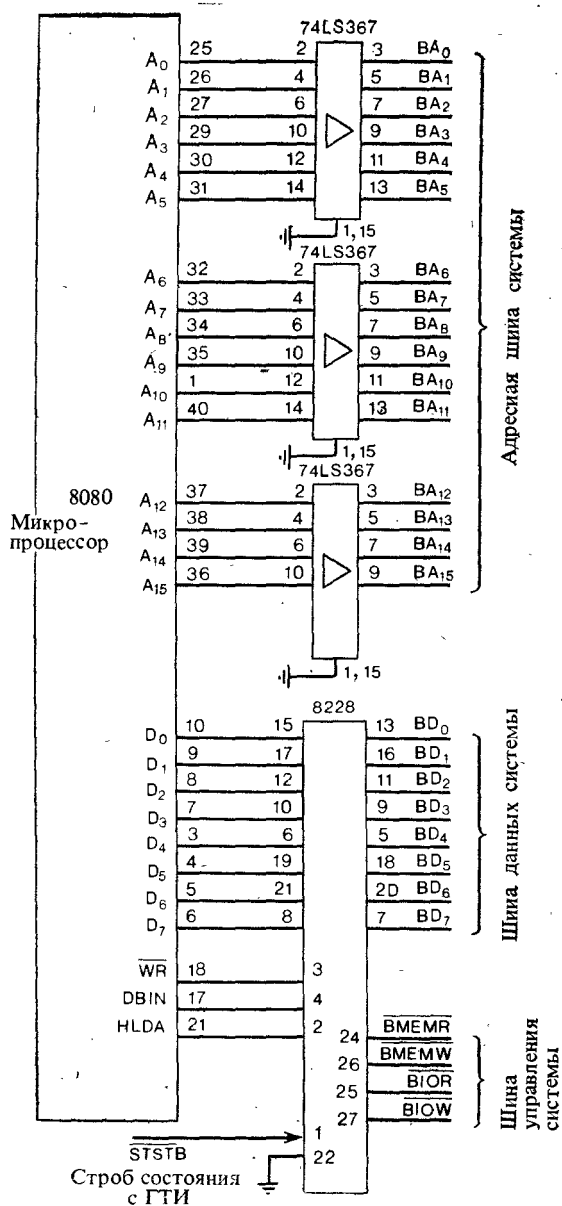


Рис. 2.32г. Схема шин: адресной, данных и управления для микропроцессора 8080.

равления, как это показано на рис. 2.31. Для любого из рассмотренных выше микропроцессоров можно реализовать архитектуру ввода-вывода по аналогии с обращением к памяти. Однако для микропроцессора 6800 такая архитектура является обязательной.

2.20. Выводы

Выше было показано, каким образом реализуется архитектура систем с 3 шинами на базе микропроцессоров 6800, Z80, 8085 и 8080. Управляющие сигналы микропроцессора использовались для генерирования сигналов шины управления и определения направления передачи информации по шине данных. При этом нет необходимости беспокоиться о синхронизации сигналов управления микропроцессора. Последний контролирует логические уровни и осуществляет синхронизацию сигналов, таких, как RD, WR, DBIN и R/W.

Из проведенного обсуждения ясно, что реализация архитектуры системы с 3 шинами для рассмотренных микропроцессоров достаточно проста. Здесь не обсуждались такие детали, как подключение памяти к системе или генерирование тактовых импульсов. Эти вопросы вынесены в последующие главы.

В завершение главы приведем полные схемы четырех рассмотренных микропроцессоров, имеющих архитектуру систем с 3 шинами. Эти схемы представлены на рис. 2.32а — 2.32г, где ясно видны три шины — адресная, данных и управления. Отметим сходство построения всех трех шин для указанных микропроцессоров.

Необходимо подчеркнуть, что рассмотренная схема построения шины управления не является обязательной. Конкретная реализация этой шины в большой степени зависит от типа памяти и других ИС, используемых в системе. Выбор представленной выше реализации шины управления сделан лишь с целью иллюстрации схожих сторон четырех рассматриваемых микропроцессоров. Однако приведенная техника построения шины управления может быть использована на практике и способствует пониманию архитектуры систем с 3 шинами.

ГЕНЕРАТОРЫ ТАКТОВЫХ ИМПУЛЬСОВ И ИНТЕРФЕЙС ПАМЯТИ ДЛЯ МИКРОПРОЦЕССОРНЫХ СИСТЕМ С 3 ШИНАМИ

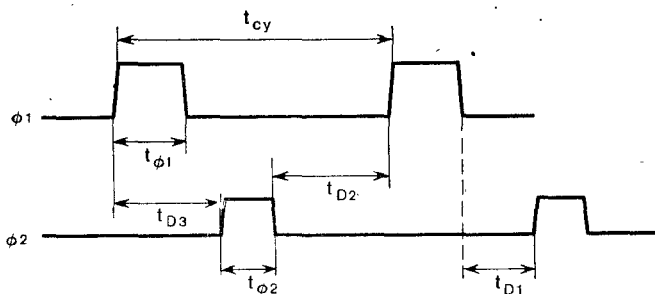
Во второй главе была рассмотрена структура микропроцессорной системы, в состав которой входят следующие три шины: адресная шина, шина данных и шина управления. Для каждого микропроцессора обсуждались требования, предъявляемые к генератору тактовых импульсов. Указывалось, что он определяет синхронизацию микропроцессорной системы. В этой главе рассмотрим вопросы генерации последовательности тактовых импульсов, т. е. покажем, как обеспечить систему необходимым генератором тактовых импульсов.

Также обсудим вопросы интерфейса памяти в системах с 3 шинами, в частности для каждого типа микропроцессора согласование электрических характеристик при использовании различных ОЗУ и ПЗУ. Кроме того, рассмотрим методы организации интерфейса в микропроцессорных системах для ОЗУ с общим входом и выходом и для ОЗУ с отдельными входом и выходом.

3.1. Генератор тактовых импульсов для микропроцессора 8080

Для микропроцессора 8080 требуются две высоковольтные (12 В) последовательности тактовых импульсов. Эти последовательности условно назовем: фаза 1 ($\phi 1$) и фаза 2 ($\phi 2$). Временные соотношения последовательностей тактовых импульсов показаны на временной диаграмме (рис. 3.1). Там же содержится перечень их временных характеристик, для которых в таблице приведены ограничения.

Временная диаграмма и ее характеристики показывают, что получение последовательностей тактовых импульсов, отвечающих заданным требованиям, представляет собой отдельную и нелегкую техническую задачу. К счастью, в рамках этой задачи для микропроцессора 8080 была разработана специальная интегральная схема. Это генератор тактовых импульсов с шифром 8224. Кроме выработки тактовой последовательности импульсов для 8080, устройство 8224 может выполнять и другие функции. Это делает его применение в микропроцессоре Intel 8080 более удобным. Дополнительные возможности генератора



Обозначение	Параметр	Значение		Единица измерения
		мин.	макс.	
t_{cy}	Период следования импульсов	0,48	2,0	мкс
t_r, t_f	Время переднего и время заднего фронта	0	50	нс
$t_{\phi 1}$	Ширина импульса $\phi 1$	60		нс
$t_{\phi 2}$	Ширина импульса $\phi 2$	220		нс
t_{D1}	Временной интервал между задним фронтом импульса $\phi 1$ и передним фронтом следующего импульса $\phi 2$	0		нс
t_{D2}	Временной интервал между задним фронтом импульса $\phi 2$ и передним фронтом следующего импульса $\phi 1$	70		нс
t_{D3}	Задержка импульса $\phi 2$ по отношению к импульсу $\phi 1$	80		нс

Примечание. Характеристики переменного тока: $T_A = 0 - 70^\circ \text{C}$, $V_{DD} = +12 \text{ В} \pm 5\%$, $V_{CC} = +5 \text{ В} \pm 5\%$, $V_{BB} = 5 \text{ В} \pm 5\%$, $V_{SS} = 0 \text{ В}$, если не оговаривается особо.

Рис. 3.1. Временная диаграмма последовательностей $\phi 1$, $\phi 2$ тактовых импульсов для микропроцессора 8080. Временные характеристики последовательностей тактовых импульсов даны в таблице. (С разрешения фирмы Intel.)

тактовых импульсов 8224 будут рассмотрены в следующих главах.

Для проектировщика, использующего систему 8080 в качестве базовой, разработка генератора тактовых импульсов не составит труда. Если в качестве системного контроллера используется микропроцессор 8080, то генератором тактовых импульсов в этом случае будет устройство 8224. Благодаря этому уменьшается объем оборудования и, конечно, увеличивается надежность системы.

Схема и спецификации генератора 8224 представлены на рис. 3.2а и 3.2б. Для его работы необходимы лишь подача пи-

Конфигурация выводов

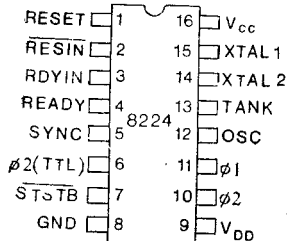
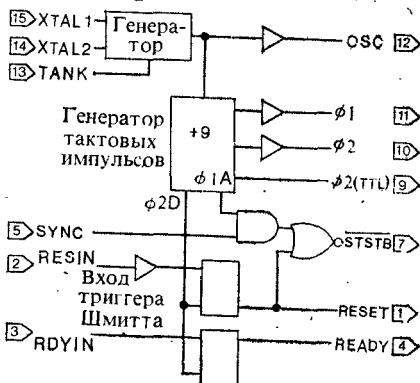


Схема устройства



RESIN	Вход «Сброс»
RESET	Выход «Сброс»
RDYIN	Вход «Готов»
READY	Выход «Готов»
SYNC	Вход «Синхронизация»
STSTB	Строб состояния Активный уровень — низкий
φ1	8080
φ2	Выходы синхрипульсов для микропроцессора

XTAL 1	Выходы для подключения кварцевого резонатора
XTAL 2	
TANK	Используется для обертон кварцевого резонатора
OSC	Выход генератора
φ2(TTL)	Фаза φ2 (TTL — уровень)
Vcc	+5 В
VDD	+12 В
GND	0 В

Рис. 3.2а. Схема и назначение выводов генератора тактовых импульсов 8224 при совместном использовании с 8080. (С разрешения фирмы Intel.)

тания от источника постоянного тока и подключение внешнего кварцевого резонатора. Требуемая частота колебаний кварцевого резонатора в 9 раз больше, чем частота импульсов, соответствующая периоду t_{cy} в последовательностях импульсов, $\phi 1$ и $\phi 2$ для микропроцессора 8080. Допустим, например, что t_{cy} равен 1 мкс, что соответствует частоте в 1 МГц. Частота колебаний резонатора должна быть равна 9 МГц, что соответствует периоду, равному девятой части t_{cy} .

На рис. 3.3 показано, как генератор 8224 подключается к микропроцессору 8080. Выходной сигнал генератора 8224 на рисунке имеет обозначение STSTB. Этот сигнал упоминался в гл. 2 как сигнал для стробирования слова состояния в фиксаторах состояний. Он подается, когда слово состояния поступает на шину данных. Если у вас нет полного представления о функциях этого стробирующего сигнала в системе, обратитесь к разд. 2.13, в котором рассматривается шина управления микро-

Характеристики генератора тактовых импульсов 8224

Обозначение	Параметр	Пределы			Единица измерения	Условие проверки
		мин.	тип.	макс.		
I_F	Входной ток			—0,25	мА	$V_F=0,45$ В
I_R	Входной ток утечки			10	мкА	$V_R=5,25$ В
V_C	Входное прямое напряжение захвата			1,0	В	$I_C=-5$ мА
V_{IL}	Низкий уровень входного напряжения			0,8	В	$V_{CC}=5,0$ В
V_{IH}	Высокий уровень входного напряжения	2,6 2,0			В В	Вход RESET Все другие входы
$V_{IH}-V_{IL}$	Напряжение на входе REDIN	0,25			мВ	$V_{CC}=5,0$ В
V_{OL}	Низкий уровень выходного напряжения			0,45	В	($\phi 1, \phi 2$), READY, RESET, STSB $I_{OL}=2,5$ мА
				0,45	В	Все другие выходы $I_{OL}=15$ мА
V_{OH}	Высокий уровень выходного напряжения $\phi 1, \phi 2$	9,4			В	$I_{OH}=-100$ мкА
	READY, RESET	3,6			В	$I_{OH}=-100$ мкА
	Все другие выходы	2,4			В	$I_{OH}=-1$ мА
I_{SC}	Выходной ток короткого замыкания (Все выходы низковольтные)	—10		—60	мВ	$V_O=0$ В $V_{CC}=5,0$ В
I_{CC}	Ток источника питания			115	мА	
I_{DD}	Ток источника питания			12	мА	

Примечание: 1. Осторожно. Выходы устройства $\phi 1, \phi 2$ не имеют защиты от короткого замыкания. 2. Характеристики постоянного тока: $T_A=0-70^\circ\text{C}$, $V_{CC}=+5,0$ В $\pm 5\%$, $V_{DD}=+12$ В $\pm 5\%$.

Рис. 3.26. Спецификации для генератора тактовых импульсов 8224. (С разрешения фирмы Intel.)

процессорной системы 8080. Мы видели, что при использовании генератора тактовых импульсов 8224 для микропроцессора 8080 нет сколько-нибудь значительных затруднений в получении кор-

ректных последовательностей тактовых импульсов $\phi 1$ и $\phi 2$. Настоятельно рекомендуем применять его для управления системой, разрабатываемой на базе микропроцессора 8080.

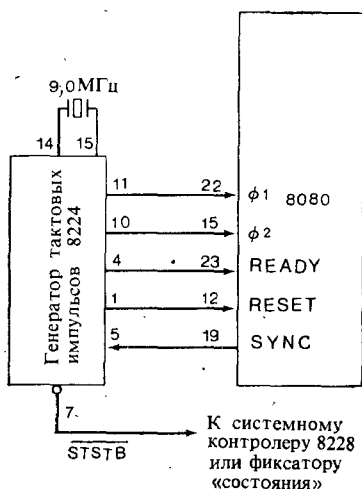


Рис. 3.3 Схема подключения генератора тактовых импульсов 8224 к микропроцессору 8080.

3.2. Генератор тактовых импульсов для микропроцессора 8085

Схема генератора тактовых импульсов микропроцессора 8085 полностью содержится в самом микропроцессоре. Достаточно просто подключить кварцевый резонатор к выводам X_1 (вы-

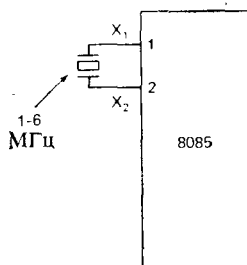


Рис. 3.4. Схема подключения кварцевого резонатора к микропроцессору 8085, предназначенного для стабилизации частоты следования тактовых импульсов. Тактовая частота микропроцессора 8085 в два раза меньше частоты колебаний кварцевого резонатора.

вод 1) и X_2 (вывод 2) микропроцессора 8085. Кварцевый резонатор может иметь любую частоту колебаний в диапазоне от 1 до 6 МГц. Эта частота делится пополам и соответствующие импульсы с периодом t_{cy} используются в микропроцессоре 8085. На рис. 3.4 показано, как путем подключения кварцевого резонатора обеспечивается синхронизация микропроцессора 8085.

3.3. Генератор тактовых импульсов для микропроцессора Z80

Для работы микропроцессора Z80 требуется одна последовательность тактовых импульсов. Такая последовательность может быть образована различными способами. На рис. 3.5 показана схема генерации синхронизирующих импульсов для микропроцессора Z80. Частота колебаний, как видно, определяется кварцевым резонатором X_1 . Однако для того чтобы эта схема работала, произведение $R_1 C_1$ должно быть численно больше,

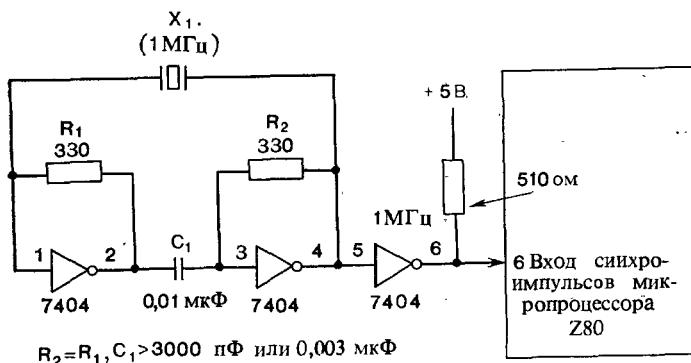


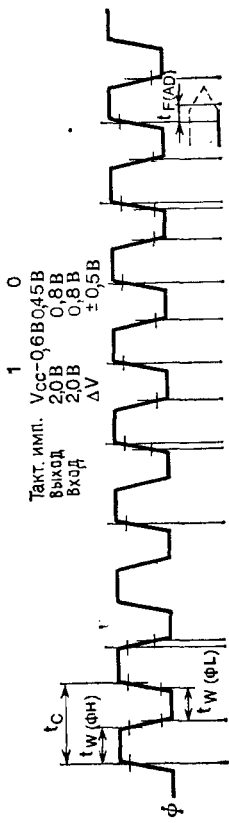
Рис. 3.5. Схема генератора тактовых импульсов для микропроцессора Z80. Выход этого генератора несовместим с ТТЛ-схемами. С помощью согласующего резистора на 510 Ом напряжение на выходе генератора тактовых импульсов увеличивается примерно до 5 В. Характеристики генератора тактовых импульсов представлены на рис. 3.6.

чем период колебаний. Допустим, требуемая частота равна 1 МГц. Ей соответствует период в $1 \cdot 10^{-6}$ с. Это означает, что величина произведения $R_1 C_1$ должна быть больше, чем $1 \cdot 10^{-6}$. Величина сопротивления R_1 должна оставаться постоянной и иметь значение приблизительно в 330 Ом. При этом значение C_1 должно быть больше или равно $1 \cdot 10^{-6} / 330$ Ф. Это же значение приблизительно равно 3000 пФ. Такая величина произведения $R_1 C_1$ обеспечивает необходимый фазовый сдвиг частоты колебаний. Спецификация характеристик генератора тактовых импульсов для микропроцессора Z80 дана на рис. 3.6.

3.4. Генератор тактовых импульсов для микропроцессора 6800

Микропроцессор 6800 подобно микропроцессору 8080 требует для своей работы две последовательности тактовых импульсов, для образования которых должны использоваться схемы, отличные от ТТЛ-схем. Эти последовательности тактовых импуль-

Если не указано особо, измерения всех временных характеристик выполнены при следующих напряжениях:



$T_A = 0^\circ\text{C}$ до 70°C , $V_{cc} = +5\text{V} \pm 5\%$

Сигнал	Обозначение	Параметр	Мин	Max	Единицы измерения	Условия проверки
ϕ	t_c	Период следования тактовых импульсов	0.4	[12]	мкс	[12] $t_c = t_w(\phi H) + t_w(\phi L), t_r + t_f$
	$t_w(\phi H)$	Ширина импульса на верхнем уровне	180		нс	
	$t_w(\phi L)$	Ширина импульса на нижнем уровне	180	2000	нс	
	$t_{r,f}$	Время переднего и заднего фронта импульса		30	нс	

Рис. 3.6. Временная диаграмма и основные временные характеристики последовательности тактовых импульсов для микропроцессора Z80. (С разрешения фирмы Zilog.)

сов имеют названия «фаза 1» и «фаза 2» соответственно. На рис. 3.7 показаны формы импульсов и даны спецификации последовательностей тактовых импульсов, предназначенных для микропроцессора 6800. Видно, что, как и для микропроцессора 8080, генерация тактовых импульсов такой формы представляет собой непростую техническую задачу. Фирма Motorola разрабо-

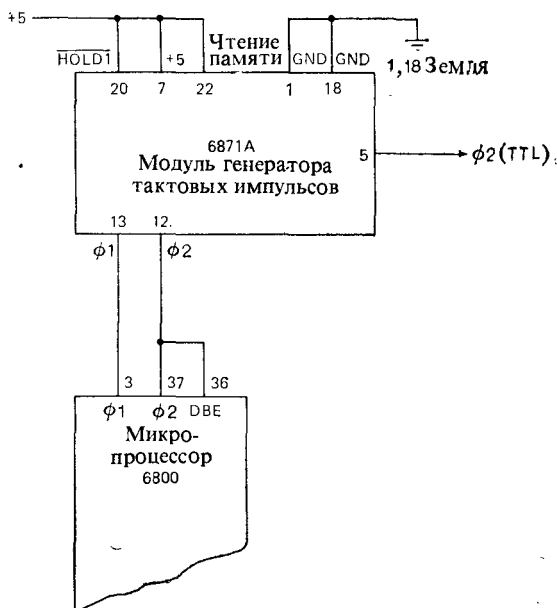


Рис. 3.8. Схема подключения генератора 6871А к микропроцессору 6800.

тала генератор тактовых импульсов, который пригоден для использования совместно с микропроцессором 6800. Такой генератор, обозначаемый 6870А или 6871А, рекомендуется использовать совместно с микропроцессором 6800 как основной компонент системы управления. На рис. 3.8 показана схема подключения генератора тактовых импульсов к микропроцессору 6800. Отметим также, что для сигналов, передаваемых по шине управления, в микропроцессоре 6800 используется задний фронт синхронизирующего импульса из фазы 2. (Если вы плохо представляете себе о чем здесь идет речь, то обратитесь к соответствующему разделу гл. 2, в котором рассматривается шина управления микропроцессора 6800.) Это означает, что последовательность «фаза 2» должна подаваться на вход устройства с ТТЛ-логикой. В микропроцессоре 6800 используется входной уровень сигнала, отличающийся от принятого в ТТЛ-схемах. Поэтому

нельзя подавать на вход микропроцессора 6800 последовательность «фаза 2», которая используется для ТТЛ-схем внутри системы, так как уровни напряжений сигналов в двух указанных последовательностях должны быть различными. Однако в генераторах тактовых импульсов 6870А и 6871А для последовательности $\phi 2$ предусмотрен специальный уровень выходных сигналов, пригодный для схем с ТТЛ-логикой, который используется для ТТЛ-схем в микропроцессорной системе 6800.

3.5. Выводы

Для обеспечения функционирования микропроцессора Z80 требуется одна последовательность тактовых импульсов, а для микропроцессоров 8080 и 6800 необходимы две несинхронизированные последовательности тактовых импульсов. Микропроцессор не будет работать нормально, если генератор тактовых импульсов функционирует неправильно. Следует отметить, что при поиске неисправностей в микропроцессорной системе целесообразно сначала убедиться в том, что все подаваемые уровни напряжения имеют требуемые значения и что все последовательности тактовых импульсов имеют необходимые временные соотношения и номинальные уровни напряжения.

3.6. Интерфейс памяти в микропроцессорных системах с 3 шинами

В этом разделе обсуждаются вопросы подключения ПЗУ и ОЗУ в микропроцессорных системах с 3 шинами. Сначала рассматривается подача сигнала по линии выбора памяти, или линии разрешения доступа к памяти, а затем организация интерфейса обычных ПЗУ и ОЗУ в микропроцессорных системах. В качестве ПЗУ выбрано устройство 2708, а в качестве ОЗУ — устройства 2102 и 2124. Выбор названных устройств обусловлен их широким распространением в настоящее время. Если знать, как подключаются эти устройства к микропроцессорной системе управления, то вполне будет понятно, каким образом подключаются другие им подобные устройства.

После того как выбрано подходящее для системы устройство памяти, составляется схема распределения памяти. О ней упоминалось в гл. 2 при рассмотрении микропроцессора 6800. Теперь подробно разберем, что подразумевается под этим понятием.

Таблица распределения памяти — это присвоение областям физически доступного пространства памяти определенных функций. Например, можно использовать область памяти с адресами 0000—03FF для ПЗУ, а область памяти с адресами 0400—07FF для ОЗУ. На формирование таблицы распределения памяти ока-

зывают влияние такие факторы, как архитектура ввода-вывода и типы устройств ОЗУ и ПЗУ. В данном обсуждении будем считать, что способ организации ввода-вывода по аналогии с обращением к памяти не используется; в качестве ПЗУ нашей системы выбрано устройство 2708, в качестве ОЗУ — устройство 2102. Таблица распределения памяти системы в этом случае будет такой, как показано на рис. 3.9. На этом рисунке представлено распределение доступного пространства памяти с диапа-

0000—03FF	ПЗУ ₁
0400—07FF	ПЗУ ₂
0800—0BFF	ПЗУ ₃
0C00—0FFF	ПЗУ ₄
1000—13FF	ОЗУ
1400—FFFF	Не используется

Рис. 3.9. Пример распределения памяти системы.

зоном адресов 0000—FFFF блоками в 1 К и для ОЗУ и для ПЗУ. Размер блока 1 К выбран из тех соображений, что физически память организуется так: $1\text{ К} \times 8\text{ ПЗУ}$ и $1\text{ К} \times 1\text{ ОЗУ}$. Причем, чтобы получить ОЗУ объемом в 1 К байт, параллельно подключаются 8 блоков ОЗУ по 1 К. Об этом будет рассказано в данной главе ниже.

В случае если в качестве ПЗУ нашей системы выбрано устройство 2716, то блоки памяти для ПЗУ будут иметь размер, равный 2 К. Это объясняется тем, что устройство 2716 представляет собой ППЗУ (перепрограммируемое постоянное запоминающее устройство) емкостью $2\text{ К} \times 8$. Очевидно, что таблица распределения памяти служит справочником, содержащим информацию о распределении доступного пространства памяти микропроцессорной системы.

Допустим, что нет необходимости использовать все пространство памяти, и поэтому для ПЗУ зарезервировано 4096 байт (или 4 К). Однако следует помнить, что, если резерв окажется слишком маленьким, при возникновении потребности в дополнительном объеме памяти для реорганизации таблицы распределения памяти в уже действующей системе могут потребоваться значительные усилия. Таким образом, если адресное пространство памяти позволяет, жесткая экономия памяти не является необходимой. Всегда стоит предусмотреть место для возможного расширения.

3.7. ПЗУ системы

Теперь рассмотрим, как микропроцессор связывается с ПЗУ электрически.

Когда микропроцессор подает на шину адреса адрес из диапазона 0000—03FF, то из таблицы распределения памяти ясно,

что для связи с ним будет открыто ПЗУ₀. Аналогичное соответствие существует для ПЗУ₁, ПЗУ₂, ПЗУ₃: Из сказанного можно заключить, что в адресной схеме системы должен протекать некоторый процесс однозначного выбора, и он реализуется подачей на линии адреса ВА₁₀—ВА₁₃ определенного кода выбора или сигнала разрешения доступа к памяти — к отдельному блоку памяти или к совокупности блоков. Например, если микропро-

Двоичные значения сигналов на линиях шины адреса, соответствующие определенным областям адресного пространства памяти

ВА ₁₃	ВА ₁₂	ВА ₁₁	ВА ₁₀	Диапазон адресов	
0	0	0	0	0000—03FF	ПЗУ ₀
0	0	0	1	0400—07FF	ПЗУ ₁
0	0	1	0	0800—0BFF	ПЗУ ₂
0	0	1	1	0C00—0FFF	ПЗУ ₃
0	1	0	0	1000—13FF	ОЗУ

Рис. 3.10. Двоичные значения сигналов на линиях шины адреса, соответствующие определенным областям адресного пространства памяти.

цессор был связан с ПЗУ₁, то на линиях адреса ВА₁₀—ВА₁₃ должны быть следующие двоичные значения 1000 соответственно. На рис. 3.10 показаны двоичные значения состояний линий адреса ВА₁₀—ВА₁₃ для нескольких диапазонов адресов.

Используя рис. 3.10, можно спроектировать комбинационную логическую схему, которая будет выдавать активный сигнал, когда адрес памяти находится в определенных пределах. А именно, когда адрес памяти относится к одному из ее блоков по 1 К, на линии выбора памяти, соответствующей этому блоку, появится сигнал. К счастью, эта проблема является классической, и она имеет простое решение. В частности, можно использовать дешифратор, выполненный в виде ТТЛ-устройства среднего уровня интеграции, например двоично-десятичный дешифратор 7442¹⁾.

На рис. 3.11 показано, как дешифратор 7442 используется в системе управления микропроцессора для дешифрирования адреса и получения сигнала выбора памяти. Изготовители полупроводниковых устройств памяти часто разрабатывают свои собственные цифровые дешифраторы, предназначенные для применения с определенными устройствами памяти. Фирма Intel, например, предлагает цифровой дешифратор 8205 «один из восьми». Его можно использовать для формирования сигналов на линиях выбора памяти.

Теперь покажем, как используются линии выбора памяти, шина адреса, шина данных и шина управления для связи мик-

¹⁾ Coffron J. W., Getting Started in Digital Troubleshooting, Reston Publishing Co., 1979.

ропроцессора и ПЗУ системы. Структурная схема связи памяти ПЗУ₀—ПЗУ₃ с шинами показана на рис. 3.12. Ниже будет объясняться принцип действия представленной на этом рисунке схемы.

Отметим, что на схеме линии адреса BA_0 — BA_9 соединяются с соответствующими адресными выводами A_0 — A_9 четырех устройств ППЗУ типа 2708. Подобный способ соединения выводов

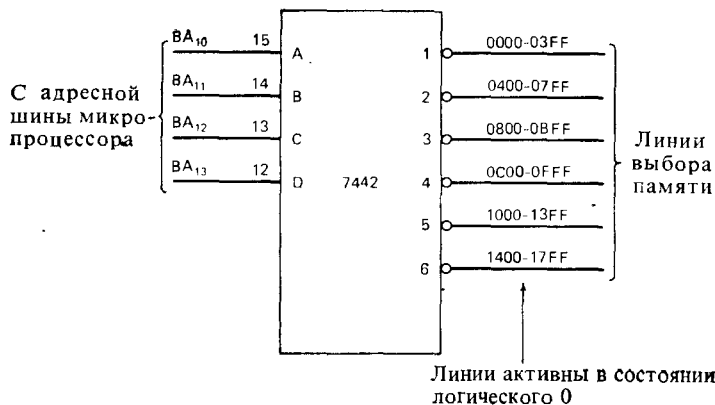


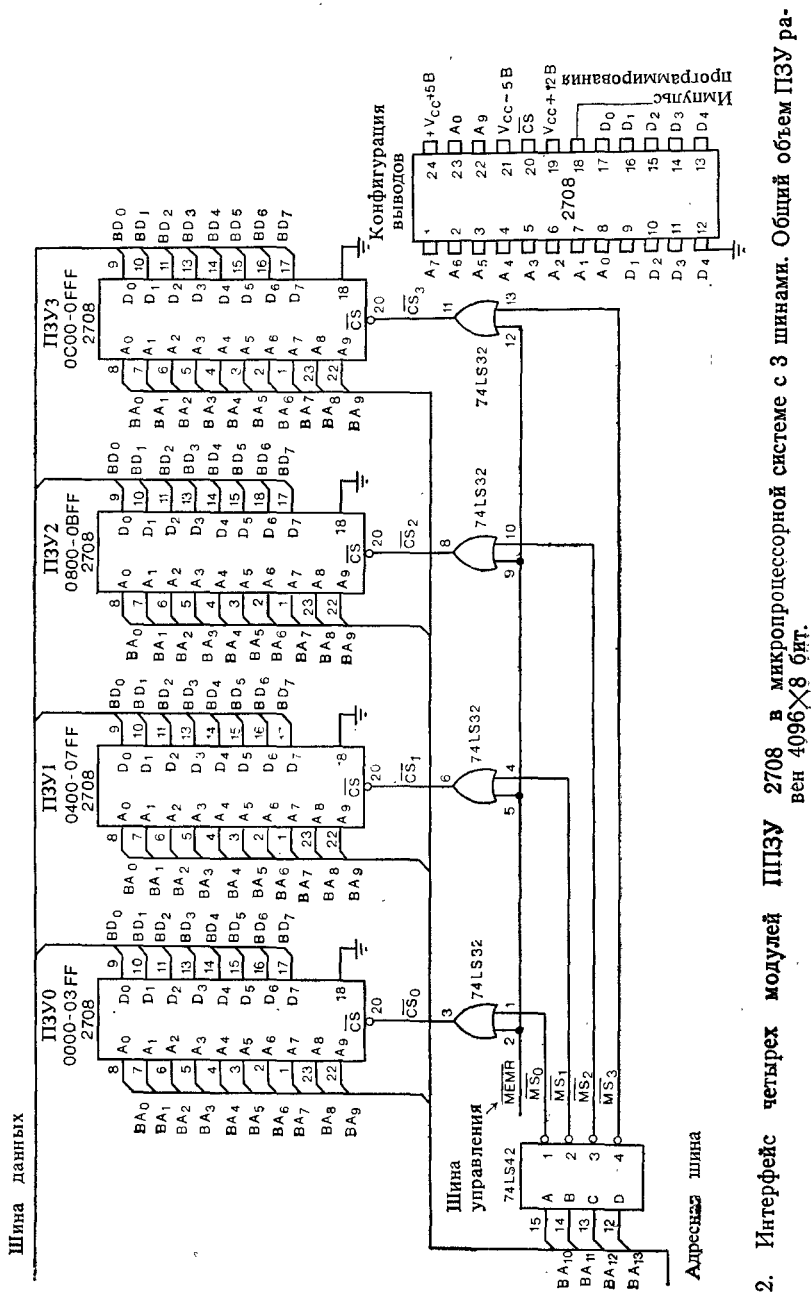
Рис. 3.11. Схема образования сигналов на линиях выбора памяти, в которой используется простой дешифратор 7442.

называют соединением в виде дейзи-цепочки. Также отметим, что выводы, используемые для выдачи данных четырех ПЗУ, связаны по принципу дейзи-цепочки с соответствующими линиями шины данных.

Вследствие того что выводы ПЗУ связаны по принципу дейзи-цепочки, попытка вывода в одно и то же время данных более чем из одного ПЗУ привела бы к непоправимым ошибкам. Такая ситуация известна как «конфликт на шине данных». «Конфликт на шине данных» происходит, когда в некоторый момент времени на линии шины данных поступает более чем один набор выходных сигналов. В действительности два выходных сигнала различных активных устройств поступают почти одновременно. Такая ситуация в системе является недопустимой.

Для предотвращения конфликта на шине данных между четырьмя модулями ПЗУ система в некоторый данный момент времени открывает только один модуль ПЗУ. Эта функция линии выбора памяти рассматривалась раньше. Для этой системы линии выбора памяти в активном состоянии имеют уровень логического 0. Однако для других систем могут быть приняты иные соглашения.

Линии выбора памяти имеют обозначения MS_0 — MS_3 . Сигналы на этих линиях являются входными сигналами для двух-



входных схем ИЛИ типа 74LS32, показанных на рис. 3.12. Когда сигнал MEMR, подаваемый по шине управления, имеет логическое значение 0, тогда на выходе схемы ИЛИ, на вход которой подается сигнал выбора памяти, имеющий логическое значение 0, будет сформирован сигнал уровня логического 0. Состояние логического 0 на выходе схемы ИЛИ теперь обеспечивает сигнал выбора кристалла в выбранном ПЗУ. Когда входной сигнал выбора кристалла ПЗУ имеет значение логического 0, выходные сигналы ПЗУ поступают на шину данных. Выходы других ПЗУ, на входах которых сигналы выбора кристалла имеют логические значения, равные 1 (пассивное состояние), находятся в состоянии высокого сопротивления. Когда выход находится в состоянии высокого сопротивления, он не будет оказывать влияния на какие-либо другие связанные с ним выходы.

Используя вывод выбора кристалла на устройстве памяти, выходы устройств памяти можно соединять по принципу дейзипечки. Например, допустим, что микропроцессорная система выполняет операцию чтения из памяти по адресу 0409₁₆. Дешифрирование сигналов, поступающих по шине адреса, приведет к выбору ПЗУ. Линия выбора MS₁ — вывод 2 устройства 74LS42 на рис. 3.12 — примет значение логического 0. Тем самым будет установлено состояние логического 0 на выходе 4 схемы 74LS32. Только одна из четырех схем ИЛИ, а именно схема ИЛИ с выводом 4, имеет логическое значение 0 на этом выводе. Теперь лишь одна эта схема подготовлена к отпиранию. Мы говорим, что «схема подготовлена к отпиранию», так как для ее отпирания должен быть подан соответствующий управляющий сигнал. Для отпирания схемы ИЛИ шина управления должна сформировать и подать на соответствующую схему ИЛИ сигнал управления чтением из памяти. Микропроцессор будет подавать сигнал управления чтением из памяти в соответствующий момент после истечения времени доступа к данным, находящимся в памяти. В это время на выводе 20 ПЗУ₁ устройства 2708 сигнал выбора кристалла будет иметь уровень логического 0. Теперь выбранные из ПЗУ данные через внутренние логические схемы ППЗУ типа 2708 поступят на шину данных.

После выполнения за соответствующее время чтения данных из ПЗУ₁ типа 2708 микропроцессор переведет шину управления в «нерабочее состояние» путем подачи сигнала, имеющего логическое значение 1, на линию управления чтением из памяти. Подобные действия выполняются каждый раз, когда микропроцессор выполняет операцию чтения данных из ПЗУ. Когда мы рассматривали действия, связанные с процессом чтения данных из ПЗУ, стали очевидными некоторые особенности этого процесса. Рассмотрим их. При этом не будем касаться проблемы получения всех синхронизирующих импульсов, так как мик-

ропроцессор сам обеспечивает синхронизацию при образовании адреса и формировании сигналов управления. При использовании микропроцессоров с 3 шинами существует сложная проблема обеспечения корректной работы всех схем дешифраторов.

Комбинационные логические схемы дешифрации сигналов, образуемые микропроцессором, должны предотвращать возможные появления конфликтов на шинах. Не исключено, что потребуются подключить микропроцессор к памяти, для которой время доступа к данным больше, чем время, выделяемое для этого действия микропроцессором. Такую задачу можно решить. Однако в большинстве случаев приходится использовать ПЗУ, для которых время доступа к данным меньше, чем время, выделяемое для этого микропроцессором. В последнем случае задача проектирования состоит в разработке простого дешифратора с использованием статических комбинационных логических схем. В гл. 6 для каждого рассматриваемого микропроцессора будет изучаться интерфейс устройств памяти, которые требуют для доступа к данным больше времени, чем выделяемое для этого микропроцессором.

После более подробного ознакомления с вопросами интерфейса ПЗУ в микропроцессорных системах будут рассмотрены некоторые популярные методы решения задачи дешифрации. Тогда, применяя испытанные методы, мы будем в состоянии быстро и уверенно проектировать простые микропроцессорные системы. Не следует думать, что методы дешифрации, изложенные в этой книге, отражают современный уровень достижений в этой области. Представленные здесь методы будут способствовать пониманию основных принципов. Эти методы пригодны для практического использования. Однако, уделив этой задаче больше внимания, с целью сокращения объема оборудования можно разработать другие рациональные методы дешифрации.

3.8. Интерфейс ОЗУ

Для выполнения различных функций в микропроцессорных системах используется несколько ОЗУ. В процессе выполнения программы ОЗУ используется для временного хранения данных и, когда используются подпрограммы, для временного сохранения адресов возврата. Далее будем рассматривать интерфейс трех типов статических ОЗУ в микропроцессорных системах с 3 шинами.

Известны два типа статических ОЗУ: ОЗУ с отдельными входом и выходом и ОЗУ с совмещенными входом и выходом. ОЗУ с отдельными входом и выходом имеют на устройстве два вывода — один для ввода данных, другой для вывода данных. ОЗУ с общим входом и выходом имеет один вывод для ввода и вывода данных. Сначала рассмотрим, как микропроцессор под-

ключается к ОЗУ с отдельными входом и выходом, затем обсудим вопрос подключения микропроцессора к ОЗУ с общим входом и выходом.

Интерфейс ОЗУ в микропроцессорных системах с 3 шинами опишем следующим образом. Сначала вспомним, что микропроцессор должен иметь возможность записывать данные в память и выводить данные из памяти. Микропроцессор выдает данные, записываемые в память, на шину данных. Данные, подлежащие выводу из памяти, выводятся на шину данных с помощью ОЗУ. Это означает, что шина данных должна иметь электрический тракт как для ввода данных в ОЗУ, так и для вывода данных из ОЗУ. Указанные связи показаны на рис. 3.13. В течение цикла записи в память входы ОЗУ связываются с шиной данных, а в течение цикла чтения из памяти выходы ОЗУ связываются с шиной данных. Заметим, что при такой схеме подключения ОЗУ данные могут передаваться по не предназначенному для них тракту, если он окажется открытым, и тогда возможна конфликтная ситуация на шине данных. Такая неприятная ситуация может произойти, потому что выходы ОЗУ соединены непосредственно с шиной данных и могут находиться в активном состоянии, в то время как микропроцессор через соответствующие входы ОЗУ попытается выполнить запись данных в память. Это показано в общих чертах на пояснительном рис. 3.14.

При использовании памяти с отдельными входом и выходом для предотвращения возможного конфликта на шине данных может быть использован следующий способ. (Заметим, однако, что рассматриваемый ниже способ предотвращения конфликта нет необходимости использовать, когда в качестве ОЗУ с отдельными входом и выходом выбраны устройства, имеющие внутренние схемы для реализации интерфейса ОЗУ с отдельными входом и выходом в микропроцессорных системах с 3 шинами.) Если понять принципы подключения ОЗУ с отдельными входом и выходом типа 2102 к микропроцессору в системе с 3 шинами, то подключение других аналогичных устройств памяти не вызовет затруднений. При использовании такого устройства памяти, как ОЗУ 2102 на время записи данных в память, необходимо отключить выходы ОЗУ от шины данных. Для этого воспользуемся устройством, которое называют буфером с тремя состояниями. Его входы соединяются с выходами памяти, а выходы — с шиной данных. Схема включения буфера показана на рис. 3.15.

Дальнейшее обсуждение будем вести на примере буфера с тремя состояниями типа 74LS367, схема включения которого показана на рис. 3.16. Это не единственное устройство, способное выполнить требуемые функции. Оно выбрано для конкретности изложения. Буфер с тремя состояниями блокируется (пе-

Рис. 3.13. Схема связи шины данных с устройством памяти, имеющим отдельные вход и выход.

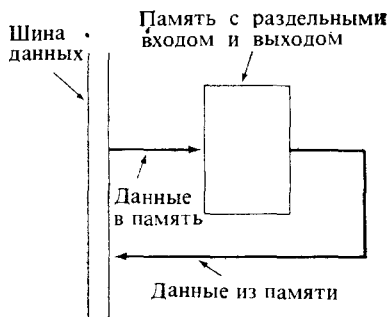


Рис. 3.14. Схема связи устройства памяти с шиной данных, при которой возможен конфликт на шине данных. Кругом на схеме отмечена точка возникновения конфликта на шине данных.

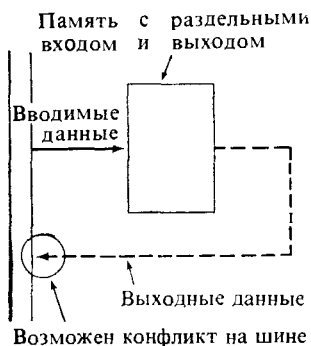
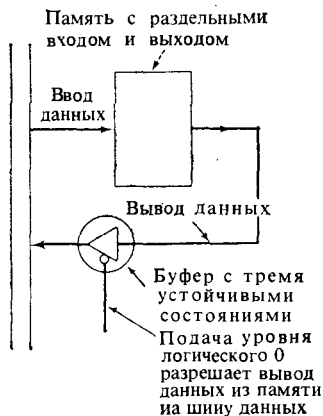


Рис. 3.15. Включение буфера с тремя состояниями на выходе ОЗУ. Когда шина данных управляется микропроцессором (запись данных в ОЗУ), выход ОЗУ отключается от шины данных. Это исключает возможность возникновения конфликта на шине данных.



реходит в состояние высокого сопротивления), когда выводы 1 и 15 находятся в состоянии логической 1. Когда микропроцессор записывает данные в ОЗУ типа 2102, на выводы 1 и 15 бу-

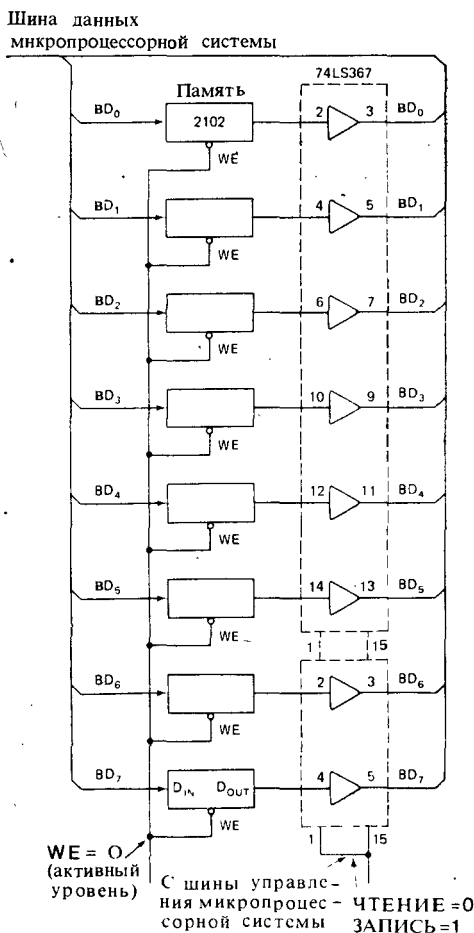


Рис. 3.16. Фрагмент схемы, обеспечивающий отключение выходов ОЗУ типа 2102 с помощью схем с тремя устойчивыми состояниями типа 74LS367.

фера с тремя состояниями подается сигнал уровня, соответствующего логическому значению 1. Во время выполнения микропроцессором любой операции записи данных в память 2102 внешние аппаратные средства должны обеспечить на указанных выводах состояние логической 1. В течение операции записи микропроцессор будет выводить данные на системную шину данных. Перед этим микропроцессор через шину управления

подает сигнал разрешения записи в память. Выход устройства 2102 на время выполнения операции записи с помощью буфера с тремя состояниями отключается от шины данных. Это необходимо для устранения возможности конфликта на шине данных. Отметим, что некоторые полупроводниковые устройства памяти с отдельными входом и выходом сами имеют буферные схемы, предназначенные для отключения их выходов от шины данных в течение операции записи в память. При использовании устройств памяти такого типа достаточно соединить входной и выходной вывод устройства памяти и затем подключить их к шине данных.

3.9. Чтение данных из ОЗУ

При выполнении операции чтения данных из ОЗУ состояния линий шины данных определяются сигналами на выходах памяти. Линии данных микропроцессора должны находиться в режиме ввода данных. Вход ОЗУ не будет воздействовать на состояние

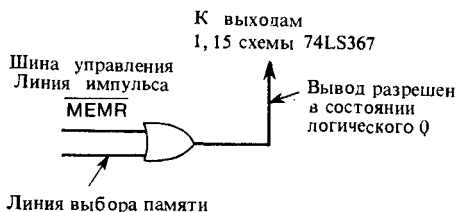


Рис. 3.17. Схема формирования сигнала, подаваемого на выходы 1 и 15 устройства 74LS367 и обеспечивающего отпирание выходов блоков ОЗУ 2102.

шины данных. Это объясняется тем, что сигналы на вход ОЗУ поступают от другого источника. Принципы функционирования ОЗУ в этом режиме почти совпадают с рассмотренными нами выше принципами действия ПЗУ.

Для формирования сигнала выбора кристалла, подаваемого на соответствующий вывод ОЗУ, линия выбора памяти и линия шины управления подключаются в соответствии со схемой, показанной на рис. 3.17. В течение времени чтения данных из ОЗУ на линии разрешения записи в ОЗУ 2102 (вывод 3) должен действовать сигнал с логическим значением 1. Когда модуль ОЗУ 2102 выбран в соответствии с кодом адреса, установленным на шине адреса, а шина управления активизирует линию записи в память, буфер с тремя состояниями открывается сигналом, подаваемым на выходы 1 и 15. Когда буфер с тремя состояниями открыт, данные из ОЗУ 2102 поступают на шину данных. На стадии чтения данных выход блока 2102 связан с соответствующей линией шины данных. После того как микропроцессор выведет данные из ОЗУ 2102, линия чтения из

памяти в системной шине управления переводится в нерабочее состояние, а буфер с тремя состояниями 74LS367 переводится в состояние высокого сопротивления. Таким образом будет осуществляться передача данных с шины данных в ОЗУ. Как видим, операция чтения данных из ОЗУ не отличается от операции чтения данных из системного ПЗУ.

3.10. Запись данных в ОЗУ

Теперь рассмотрим порядок выполнения операции записи данных в память. Чтобы записать данные в устройство памяти типа 2102, необходимо выполнить следующие действия:

1. Подать на адресные линии адрес памяти.
2. Подать данные на линии ввода данных ОЗУ.
3. На линию разрешения записи в память подать сигнал, устанавливающий на линии низкий уровень напряжения.

Итак, адрес будет подан на шину адреса микропроцессорной системы; линия выбора памяти будет активизирована после дешифрирования кода адреса на адресной шине; микропроцессор будет выдавать информацию на шину данных. Данные готовы для ввода в память.

Однако ОЗУ на этой стадии не готово к выполнению операции, так как на линию выбора кристалла не подан соответствующий сигнал. Чтобы записать данные в память, необходимо ее открыть. Но отпирание памяти фактически заключается в отпирании выходов памяти. Это обуславливает конфликт между данными на выходе микропроцессора, т. е. данными, записываемыми в ОЗУ, и данными, выводимыми из ОЗУ. Чтобы предотвратить этот возможный конфликт, между выходными линиями памяти и шиной данных включается буфер с тремя состояниями типа 74LS367. Теперь, когда подан сигнал разрешения записи и память открыта, выход ее для избежания конфликта с передаваемыми из микропроцессора в ОЗУ данными блокируется посредством устройства 74LS367. Соответствующая последовательность действий показана на временной диаграмме (рис. 3.18). Ясно, что открыть ОЗУ необходимо перед выполнением операции чтения или записи данных. Однако линия разрешения записи в память находится в рабочем состоянии только тогда, когда выполняется запись данных в ОЗУ. Логическая схема для формирования необходимых управляющих сигналов приведена на рис. 3.19.

Мы рассмотрели, как для одного устройства памяти в микропроцессорных системах с 3 шинами выполняются операция чтения данных из памяти и операция записи данных в память. Но шина данных микропроцессорной системы имеет 8 разрядов. Это означает, что одновременно должно быть прочитано или записано 8 разрядов. Так как применяемые нами блоки па-

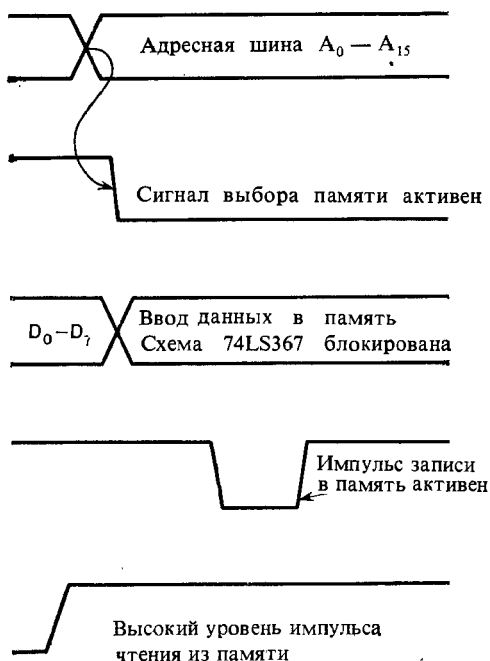


рис. 3.18. Временная диаграмма сигналов, используемых системой во время выполнения операции записи в память.

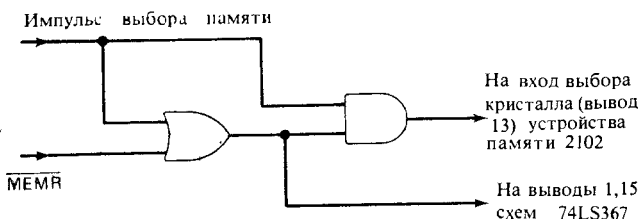


рис. 3.19. Использование линии выбора памяти и шины управления для получения сигналов управления ОЗУ.

мемории имеют одну входную линию и одну выходную линию, необходимо одновременно читать из восьми отдельных блоков ОЗУ и также одновременно записывать данные в восемь блоков ОЗУ. Это достигается посредством синхронного выполнения всех функций управления ОЗУ, т. е. подачи адреса, формирования сигналов разрешения записи и выбора кристалла. Все выходы управления восьми отдельных модулей ОЗУ соединены в виде дейзи-цепочки. Линии ввода данных соединяются с соответствующими линиями шины данных. Выходы ОЗУ подключаются к буферу с тремя состояниями типа 74LS367. Выход буфера 74LS367 подключается к входу, предназначенному для

ввода данных, того же самого модуля ОЗУ. Таким образом организуется память в виде статического ОЗУ $1\text{K} \times 8$. Функциональная схема такого ОЗУ показана на рис. 3.20.

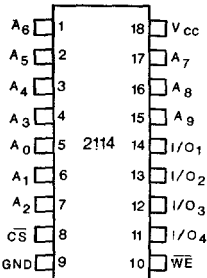
3.11. Интерфейс ОЗУ с общим входом и выходом

Интерфейс ОЗУ, имеющего общий вход и выход, подобен рассмотренному выше интерфейсу ОЗУ с разделенными входом и выходом. Однако ОЗУ с общим входом и выходом имеет один вывод, используемый и для ввода данных, и для вывода данных. Это не приводит к путанице внутри блока памяти, так как очевидно, что в один и тот же момент времени можно либо читать данные из памяти, либо вводить данные в память. Устройство памяти никогда не может выполнять эти две функции одновременно. Дальнейшее рассмотрение будем вести на примере статического ОЗУ $1\text{K} \times 4$ с общим входом и выходом типа 2114.

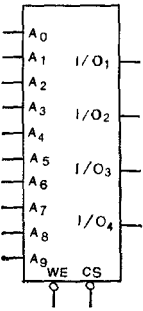
При выполнении записи данных в ОЗУ 2114 общий вывод ОЗУ действует как вход. В течение времени записи выходы ОЗУ 2114 блокируются посредством внутренних схем ОЗУ. На рис. 3.21 представлены некоторые технические сведения об ОЗУ 2114. Прежде чем микропроцессор начнет подавать данные на общие выводы памяти, она должна быть подготовлена к работе, т. е. должен быть выбран соответствующий блок памяти. Чтобы избежать конфликта на внутренних выходных линиях памяти при вводе данных в ОЗУ, когда линия разрешения записи имеет уровень логического 0, внутренние выходы ОЗУ блокируются с помощью внутренних схем. Указанные внутренние выходы переводятся при этом в состояние высокого сопротивления, что предотвращает возникновение какого-либо конфликта на шине данных. Это означает, что во время выполнения операции записи данных в устройство памяти 2114 в состоянии логического 0 должна находиться как линия разрешения записи, так и линия выбора кристалла. Это показано на временной диаграмме, изображенной на рис. 3.22.

При считывании данных из ОЗУ 2114 линия выбора кристалла должна иметь состояние логического 0, между тем как линия разрешения записи поддерживается в состоянии логической 1. Это достигается с помощью внешних схем. Когда линия выбора кристалла переходит в состояние логического 0, данные из ОЗУ подаются на общий вход-выход. (В течение времени выполнения операции чтения из памяти общие выводы устройства 2114 играют роль выходов.) Соответствующая схема, предназначенная для формирования требуемых управляющих сигналов, представлена на рис. 3.23. Схема организации памяти, построенной из модулей ОЗУ типа 2114 для микропроцессорной системы с 3 шинами, представлена на рис. 3.24. Как

Конфигурация выводов



Обозначение на схемах



Назначение выводов

A ₀ -A ₉	Входы адреса	V _{cc}	Питание
WE	Разрешение записи	GND	Земля
CS	Выбор кристалла		
I/O ₁ -I/O ₄	Ввод-вывод данных		

Схема устройства

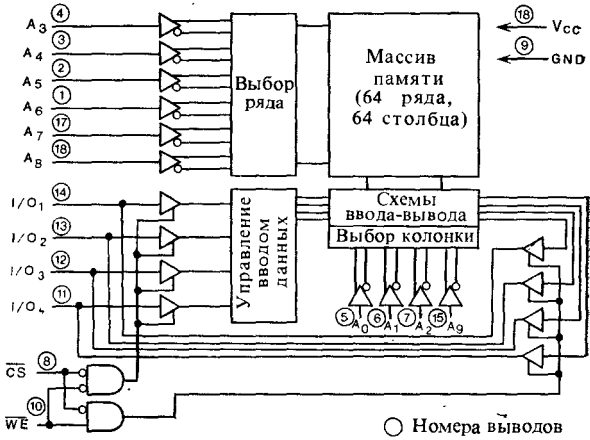


Рис. 3.21. Структурная схема и назначение выводов модуля статического ОЗУ 1024×4 типа 2114. (С разрешения фирмы Intel.)

Разрешение записи в память

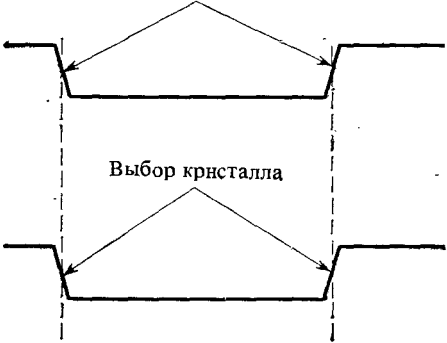


Рис. 3.22. Временная диаграмма сигнала выбора кристалла и сигнала разрешения записи. Чтобы избежать конфликта на шине данных при использовании ОЗУ с общим входом и выходом, сигнал с логическим значением 0 должен одновременно подаваться на линию выбора кристалла и на линию разрешения записи.

видно, для реализации статической памяти $1\text{К} \times 8$ требуются только два модуля типа 2114. Вспомним, что при использовании модулей статического ОЗУ типа 2102 для реализации ОЗУ объ-

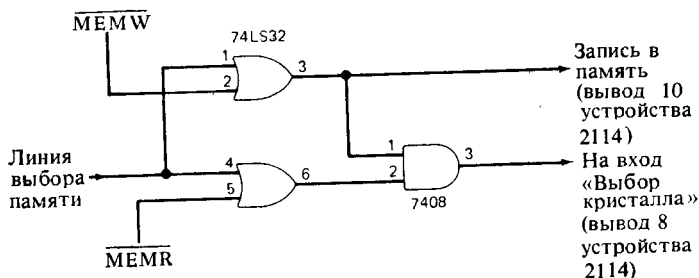


Рис. 3.23. Схема формирования сигналов управления памятью с использованием шины управления и линии выбора памяти.

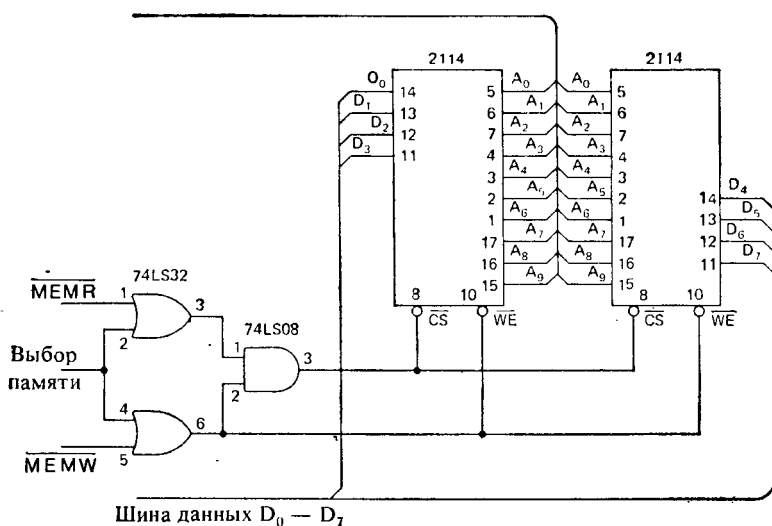


Рис. 3.24. Схема ОЗУ 1024×8 бит, содержащая два блока ОЗУ 2114.

емом $1\text{К} \times 8$ потребовалось 8 блоков памяти. Этим объясняется большая популярность модуля 2114 в небольших микропроцессорных системах.

3.12. Микропроцессор как системный контроллер

Перечислим основные аппаратные средства обычных микропроцессорных систем. К ним относятся: генератор тактовых импульсов, ПЗУ и ОЗУ, шина адреса, шина управления, шина данных. Конечно, для управления системой необходимо доба-

вить некоторые устройства ввода и вывода, которые лишь дополняют описанную нами основную систему. В гл. 4 подробно рассматриваются различные типы устройств ввода-вывода, служащих неотъемлемой частью типичных микропроцессорных систем. Теперь обсудим некоторые условия работы аппаратных средств, являющиеся необходимыми для корректной работы микропроцессора. Будем полагать, что в нормальном цикле работы микропроцессорной системы: 1) не возникают запросы на прерывания, 2) не используется режим прямого доступа к памяти, 3) не допускаются паузы и ожидания. Эти положения не внесут коренных изменений, поскольку для многих микропроцессорных систем работа в таком режиме является вполне обычной. Когда микропроцессор работает в таком режиме, он является системным контроллером. Причем он не прекратит выполнение программы до тех пор, пока питание не будет выключено.

В следующих разделах этой главы рассмотрим, каким образом должен быть подключен каждый из четырех микропроцессоров, чтобы он работал в указанном режиме.

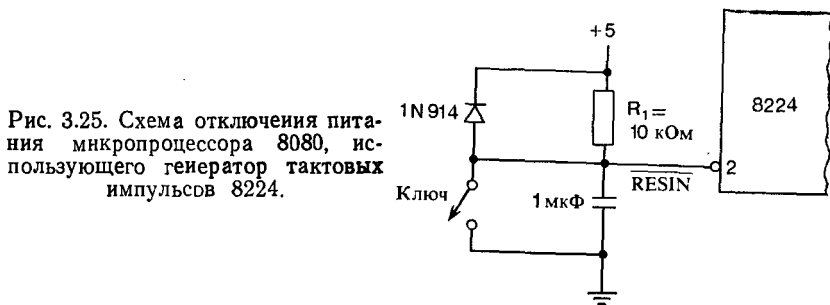
3.13. Подготовка микропроцессора 8080 для работы в режиме системного контроллера

Для подготовки микропроцессора 8080 к работе в режиме системного контроллера необходимо вывод 13 (вход сигнала HOLD) соединить с корпусом, т. е. обеспечить на нем уровень напряжения, соответствующий логическому 0. Это исключит возможность использования в системе канала прямого доступа к памяти. Вход INT (вывод 14) также должен быть соединен с корпусом. Тем самым микропроцессор 8080 будет защищен от прерываний. Затем вход RDYIN генератора тактовых импульсов необходимо соединить с источником питания так, чтобы на этом входе был уровень, соответствующий логическому значению 1. Это даст возможность избежать наступления пауз и ожиданий во время режима нормального функционирования микропроцессора. В итоге описанные действия приведут к тому, что микропроцессор будет работать с соблюдением сделанных выше оговорок. Этого удалось достигнуть, соединив выводы, на которые обычно подаются сигналы разрешения рассмотренных функций, к выводам, на которых всегда поддерживаются уровни напряжений, обеспечивающие блокировку соответствующих входов.

3.14. Установка начального состояния микропроцессора 8080

После включения питания центральный процессор должен начинать выполнение программы каждый раз с команды, расположенной в ячейке с определенным адресом, а не в какой-либо

произвольной ячейке. Для этого нужно выполнить начальную установку микропроцессора. Такая начальная установка осуществляется при первом включении микропроцессора, а также в любое время, когда потребуется вернуть микропроцессор к началу выполнения системной программы, всегда с одной и той же определенной ячейки памяти. Чтобы выполнить функции начальной установки микропроцессора 8080, к входу RESIN (вывод 2) генератора тактовых импульсов 8224 подключаются



элементы, соединенные в соответствии со схемой, показанной на рис. 3.25. После запуска микропроцессор 8080 начинает выполнять программу с команды, имеющей адрес 0000. Тем самым всегда можно принудительно начать выполнение программы с некоторой фиксированной начальной команды.

Из рис. 3.25 видно, что RC-цепочка обеспечит подачу сигнала «сброса». Когда подается напряжение V_{CC} , конденсатор находится в разряженном состоянии. После приложения напряжения конденсатор будет заряжаться до напряжения V_{CC} через резистор R_1 . Когда напряжение достигнет некоторого определенного значения, выполнение команды «сброс» завершится и система начнет выполнение программы с адреса 0000.

Ключ, показанный на рис. 3.25, позволяет выполнять «сброс» системы в любое время. Нажатие ключа приводит к разрядке конденсатора, и микропроцессор будет находиться в исходном состоянии до тех пор, пока напряжение на конденсаторе не достигнет требуемого значения. Устройство 8224 — генератор тактовых импульсов — имеет на входе триггер Шмитта, который управляет напряжением между конденсатором и резистором и в то же самое время дает маленькую нагрузку RC-цепочке.

На рис. 3.26 изображена полная микропроцессорная система 8080 с обозначениями всех выводов и указанием необходимых межсоединений, которая позволит микропроцессору действовать как системному контроллеру в соответствии с описанными выше предположениями.

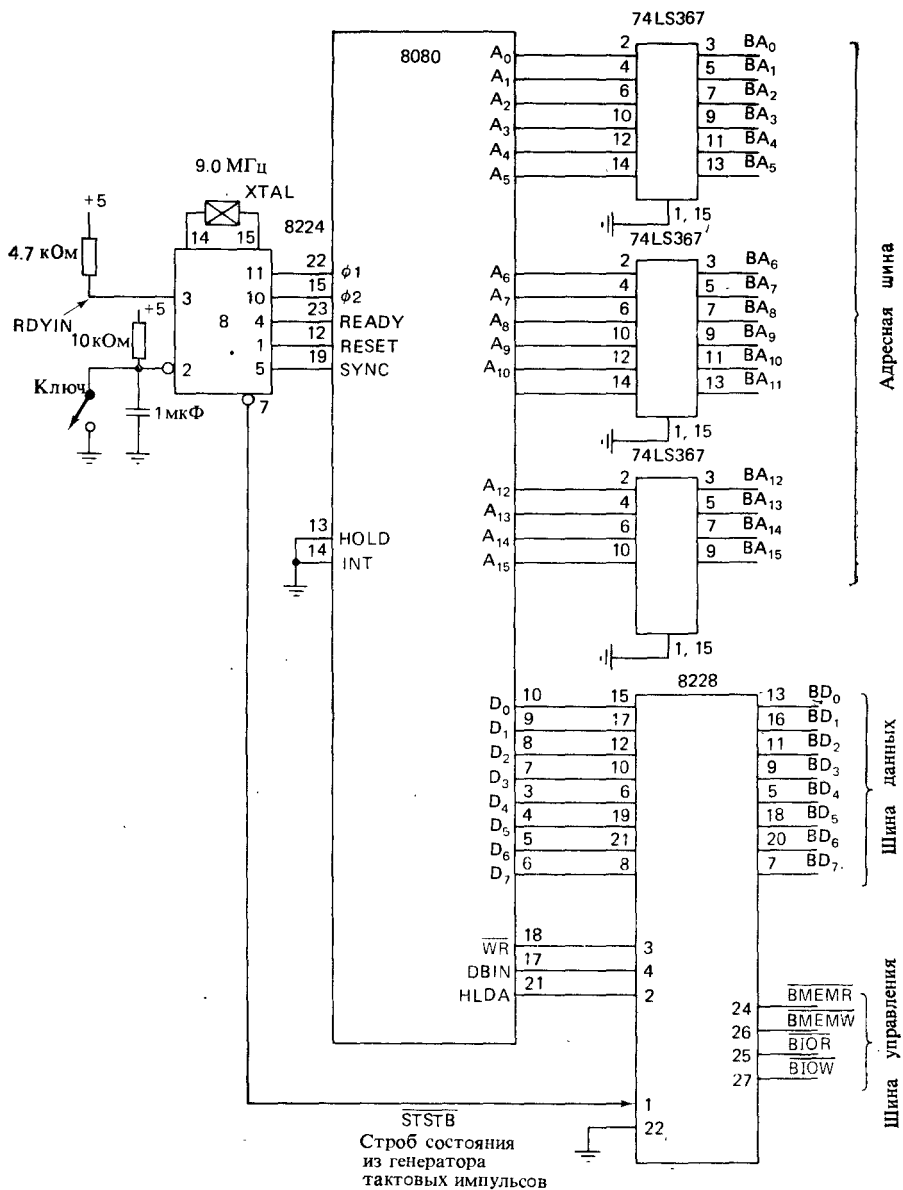


Рис. 3.26. Схема использования микропроцессора 8080 в качестве контроллера системы.

3.15. Подготовка микропроцессора 8085 для работы в режиме системного контроллера

Рассмотрим теперь, каким образом микропроцессор 8085 подготавливается к работе в режиме системного контроллера. Для этого соединим вход HOLD (вывод 39) с корпусом. Это защитит память от прямого доступа. Потом подадим на вход READY (вывод 35) напряжение уровня, соответствующего логической 1. Это предотвратит паузы во время нормального выполнения программы. Наконец, соединим выводы TRAP (вывод 6), RST 7.5 (вывод 7), RST 6.5 (вывод 8), RST 5.5 (вывод 9) и INTR (вывод 10) с корпусом. Это исключит возможность прерывания микропроцессора 8085 во время нормального выполнения программы. Названные режимы работы микропроцессора 8085 будут рассматриваться ниже при обсуждении способов организации прерываний для различных микропроцессоров.

Для обеспечения начальной установки микропроцессора 8085 к входу «сброс» (вывод 36) подключаются элементы, образующие такую же схему, как и схема, подключаемая к входу «сброс» генератора тактовых импульсов 8224. Включенный таким способом микропроцессор будет работать как контроллер системы, в котором не допускается режим прямого доступа в память, запрещаются прерывания и состояния ожидания. Соответствующая полная схема системы 8085, организованной по рассмотренному способу, представлена на рис. 3.27.

3.16. Подготовка микропроцессора Z80 для работы в режиме системного контроллера

Рассмотрим микропроцессор Z80 с точки зрения использования его как системного контроллера. Чтобы исключить возможность прямого доступа в память, соединим вывод 25, на который подается сигнал BUSRQ, с источником питания и тем самым обеспечим на этом входе уровень напряжения, соответствующий логической 1. Также обеспечим уровень, соответствующий логической 1, на выводе 24, который предназначен для приема входного сигнала WAIT. Тем самым микропроцессор будет защищен от пауз во время нормального выполнения программы. Наконец, на выводы 16 и 17, обозначенные через INT и NMI соответственно, подадим уровень логической 1. Это предотвратит возможность наступления прерываний при работе микропроцессора.

3.17. Начальная установка микропроцессора Z80

Для установки начального состояния микропроцессора Z80 может быть использована схема, представленная на рис. 3.28. Эта схема действует аналогично схеме начальной установки,

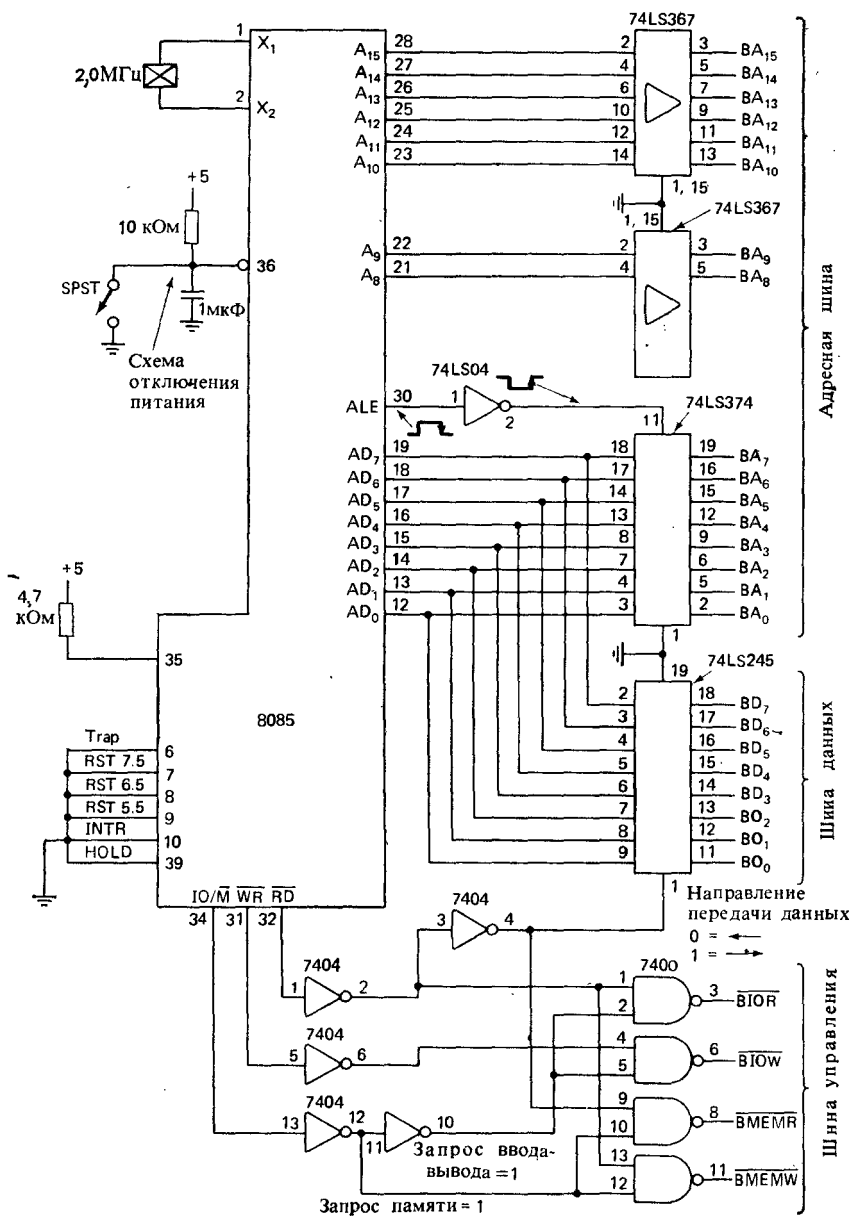
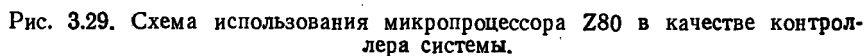
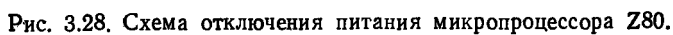


Рис. 3.27. Схема использования микропроцессора 8085 в качестве контроллера системы.



предназначенной для микропроцессоров 8080 и 8085. Как и в случае указанных микропроцессоров, Z80 после установки в начальное состояние начинает выполнять программу с команды, находящейся по адресу 0000. На рис. 3.29 представлена полная схема микропроцессорной системы Z80, спроектированной для работы в качестве системного контроллера.

3.18. Подготовка микропроцессора 6800 для работы в режиме системного контроллера

Теперь сосредоточим внимание на некоторых особенностях схемы включения микропроцессора 6800 в качестве системного контроллера. На вход HALT (вывод 2) подадим уровень, соответствующий логическому значению 1. Это предотвратит возможность выполнения операции прямого доступа в память. Обеспечим также уровень логической 1 на выводах 4 и 6, на которые подаются входные сигналы IRQ и NMI. Благодаря этому в микропроцессоре 6800 не будут возникать прерывания.

В микропроцессоре 6800 не могут возникать паузы во время нормального выполнения программы. Это достигается благодаря «растягиванию» сигналов генератора тактовых импульсов. Более подробно рассмотрим этот вопрос в гл. 6.

Подадим также вторую фазу тактовых импульсов на вход DBE (сигнал отпираания шины данных). Когда в последовательности «фаза 2» появляется уровень логической 1, шина данных обеспечивает передачу данных в микропроцессорной системе.

3.19. Начальная установка микропроцессора 6800

Схема начальной установки микропроцессора Z80, рассмотренная нами выше, может использоваться и для начальной установки микропроцессора 6800. В этом случае она выглядит, как показано на рис. 3.30. Напомним, что после выполнения начальной установки микропроцессор 6800 не начинает выполнение программы с ячейки, имеющей адрес 0000.

После выполнения начальной установки микропроцессор подает на адресную шину адреса FFFE и FFFF. Содержимое ячейки с адресом FFFE представляет собой младшие восемь разрядов формируемого адреса. По адресу FFFF находятся восемь старших разрядов формируемого адреса, которые рассматриваются микропроцессором как некоторые данные. Полный 16-разрядный адрес получается из слов, содержащихся в указанных ячейках. Таким образом, ячейки памяти FFFE и FFFF содержат фактический адрес, с которого начинается выполнение программы после начальной установки микропроцессора 6800.

Эта особенность микропроцессора 6800 обыкновенно используется для адресации в область старших адресов имеющегося в распоряжении адресного пространства ПЗУ (64К). Теперь ясно, что в ПЗУ должен находиться адрес, который должен при запуске микропроцессора разместиться на шине данных. Рассмотрим пример. Предположим, что в ячейке памяти с адресом FFFE содержится код 63, а в ячейке памяти с адресом FFFF — код 7F. Таким образом, когда микропроцессор устанавливается в начальное состояние, программа восстановления системы начнет свое выполнение с адреса 7F63, тогда как другие микропроцессоры после «сброса» начинают работать по программе, выбирая первую команду из ячейки с адресом 0000.

Полная структурная схема микропроцессора 6800, используемого в качестве системного контроллера, представлена на рис. 3.31.

3.20. Выводы

В этой главе были рассмотрены некоторые детали реализации микропроцессорных систем с 3 шинами, построенных на микропроцессорах 6800, 8080, 8085 и Z80. Были показаны способы подключения ПЗУ и ОЗУ к микропроцессорам указанных типов. Наконец, были представлены полные системные проекты применения каждого из четырех типов микропроцессоров в качестве контроллеров микропроцессорных систем.

В оставшейся части книги будут рассмотрены некоторые другие особенности микропроцессорных систем и вопросы усовершенствования изученных нами типовых вариантов построения систем. Однако первый шаг уже сделан. Важно понимать принцип построения типовых вариантов микропроцессорных систем, прежде чем приниматься за изучение более сложных дополнительных схем и возможностей. Рассмотренное по существу является «скелетом» системы. Однако не следует впадать в заблуждение и считать такую систему бесполезной. В действительности справедливо обратное.

Дело в том, что системы, построенные на основе таких скелетов, используются повсеместно. Иногда забывают, что для того чтобы оправдать применение микропроцессора в проекте системы, нет необходимости использовать все его возможности. Также иногда забывают, что используемые системы не нуждаются в приукрашивании. В настоящее время микропроцессор является очень дешевым устройством и в большинстве случаев делается по совершенно простому схемному проекту. Нет необходимости быть высококвалифицированным проектировщиком систем или инженером-проектировщиком, чтобы проектировать и эксплуатировать микропроцессорные системы. Впервые приступая к проектированию микропроцессорной системы, испыты-

ваешь естественную тревогу. В следующих главах будут продемонстрированы методы дальнейшего развития исходной типовой микропроцессорной системы. Эти методы могут быть предложены лицам, которые работают в режиме экономии и которые не имеют тысяч долларов для расхода на разработку оборудования. Рассматриваемые нами методы будут интересны для лиц, которые заняты «кустарной» разработкой проекта микропроцессорной системы, и для специалистов, работающих в промышленности.

В дальнейшем мы предполагаем обсудить фазу проектирования небольшой системы и фактическое прохождение всех этапов проектирования систем микропроцессорного управления. Конечно, чтобы успешно решить поставленную задачу, необходимо иметь ясное представление о том, как микропроцессор связывается с другим оборудованием системы. Эта глава содержала начальные сведения; каждая следующая глава, по мере того как наши знания и понимание будут расти, будет раскрывать дополнительные подробности.

Усвоив материал, содержащийся в этой главе, и рассмотрев детали, относящиеся к частным системам, представленным в следующей главе, мы добьемся твердого понимания основ. Таким образом, в указанных главах будет представлена информация, которой необходимо овладеть, прежде чем изучать вопросы практического использования микропроцессоров, действующих в реальной обстановке. На протяжении этой книги для пояснения принципов организации микропроцессорных систем используются схемы устройств, наряду с которыми часто приводятся и их полные структурные схемы. Мы нигде не полагаемся на воображение читателей.

ИНТЕРФЕЙС УСТРОЙСТВ ВВОДА-ВЫВОДА В МИКРОПРОЦЕССОРНЫХ СИСТЕМАХ С 3 ШИНАМИ

В гл. 3 рассматривались вопросы реализации аппаратного обеспечения микропроцессорных систем с 3 шинами, базирующихся на микропроцессорах 8080, 8085, Z80 и 6800. В частности, там были представлены обычные микропроцессорные системы с ПЗУ и ОЗУ. Однако мы не касались вопросов применения в системе устройств ввода-вывода.

В данной главе в рамках систем с 3 шинами рассматривается интерфейс некоторых типичных устройств ввода-вывода. При изучении функционирования каждого из четырех микропроцессоров в качестве контроллеров системы будут представлены существующие аппаратные и программные средства, соответствующие рассматриваемым типам микропроцессоров. Из устройств ввода-вывода остановимся на двух следующих: 1) клавишный пульт; 2) цифровой индикатор. Эти устройства ввода-вывода выбраны потому, что им присущи наиболее типичные функции и каждый, кто имеет дело с микропроцессорами, неизбежно воспользуется ими.

Другим существенным основанием для выбора этих устройств ввода-вывода является то, что некоторые читатели, возможно, захотят использовать излагаемые здесь идеи при конструировании небольших микропроцессорных систем. В частности, при конструировании микропроцессорной системы для конкретного применения клавишный пульт и цифровой индикатор могут быть включены в состав системы. В этом случае, даже если это первая попытка проектирования и применения микропроцессора в качестве контроллера, цифровой индикатор и простое нажатие на клавиши обеспечат вам взаимодействие с системой.

Такое взаимодействие позволит выявить важные принципы, которые окажутся полезными в ряде других применений микропроцессорных систем. Материалы данной главы будем излагать в следующей последовательности:

1. Обсуждение основных принципов построения клавишного пульта.

2. Вопросы реализации 25-клавишного пульта и соответствующего аппаратного обеспечения. (При необходимости клавиатура может быть расширена.)

3. Обсуждение основных принципов построения цифрового индикатора.

4. Вопросы реализации 6-разрядного цифрового индикатора и соответствующего аппаратного обеспечения. (Число разрядов индикатора может быть увеличено.)

5. Покажем, как клавишный пульт и цифровой индикатор подключаются к микропроцессорной системе с трехшинной архитектурой.

6. Рассмотрим программу, обеспечивающую работу клавишного пульта и цифрового индикатора. Кроме фактически существующего программного обеспечения для микропроцессоров 8080, 8085, Z80 и 6800, будут представлены схемы, соответствующие изучаемым микропроцессорным системам.

4.1. Программное обеспечение клавишного пульта

Теперь обсудим принципы «программного обеспечения клавишного пульта». Понятие «программное обеспечение клавишного пульта» используется потому, что выполнение действий, обусловленных нажатиями клавиши, и управление осуществляются программными средствами. Этот тип клавишного пульта отличается от клавишного пульта, реализованного аппаратно, когда цифровое логическое устройство управляет клавиатурой и выполняет определенные аппаратные функции, соответствующие нажатой клавише.

Сначала в общих чертах обсудим функцию клавишного устройства ввода. Часто клавишный пульт представляет собой матрицу из однополюсных нажимных нефиксируемых кнопок, которые в нормальном положении разомкнуты и переключаются мгновенно. Схематическое изображение ключа такого типа показано на рис. 4.1. Ключи располагаются в виде матрицы, которая может иметь любой размер. Будем вести рассмотрение на примере матрицы 5×5 , представленной на рис. 4.2. Очевидно, что каждый ключ однозначно определяется указанием номера строки и номера столбца, на пересечении которых находится выбранный ключ. Столбцам соответствуют обозначения $C_0—C_4$, а строкам — $R_0—R_4$. Некоторый ключ, имеющий определенную позицию в пределах матрицы программно-управляемой клавиатуры, не снабжается специальными аппаратными средствами.

Благодаря надлежащему программированию микропроцессора можно программно определять позицию нажатой кнопки и потом выполнить раздел программы, однозначно соответствующий частной позиции кнопки (ключа) в матрице клавиатуры. Это означает, что функциональное назначение любой кнопки можно изменить посредством простого изменения программного обеспечения микропроцессора. Этот принцип иллюстрируется блок-схемой, показанной на рис. 4.3.

Понимая общие принципы реализации программируемой клавиатуры, мы получаем в распоряжение «строительный блок», который пригоден для многих приложений микропроцессорных систем.



Рис. 4.1. Схематическое изображение однополюсного нормально разомкнутого ключа.

Рассмотрение принципов реализации клавишного пульта связано с решением следующих проблем:

1. Аппаратное обеспечение, требуемое для реализации клавишного пульта.
2. Программное обеспечение для управления клавишным пультом.
3. Программное обеспечение для определения замыкания ключа.
4. Программное обеспечение для определения размыкания ключа.

4.2. Аппаратные средства, необходимые для реализации клавишного пульта

Основным предметом нашего рассмотрения будут аппаратные средства, необходимые для реализации 5×5 -матрицы ключей клавишного устройства ввода. Эти аппаратные средства показаны на схеме, представленной на рис. 4.4. Для реализации должны выбираться такие интегральные схемы, которые обеспечивают минимальное общее число корпусов. Однако в данной книге мы не будем придерживаться такого критерия. Нам представляется более важным разобраться в основной идее, понять которую будет легче в процессе рассмотрения уже существующих аппаратных средств. Одни и те же функции могут быть реализованы различными аппаратными средствами. Некоторые интегральные схемы, на которые ссылаются в учебниках, приобрести трудно. Интегральные схемы, используемые для проектирования в данной книге, в настоящее время широко распространены. Тем не менее постоянно разрабатываются новые интегральные схемы, обладающие новыми функциями. Поэтому необходимо все время следить за сообщениями в литературе, чтобы своевременно знать о появившихся интегральных схемах. Их использование позволит увеличить надежность микропроцессорных систем и приведет к уменьшению общего числа корпусов. Еще раз подчеркнем, что прежде всего надо понимать основные принципы реализации, и, чтобы донести их до читателя, разберем характерные примеры.

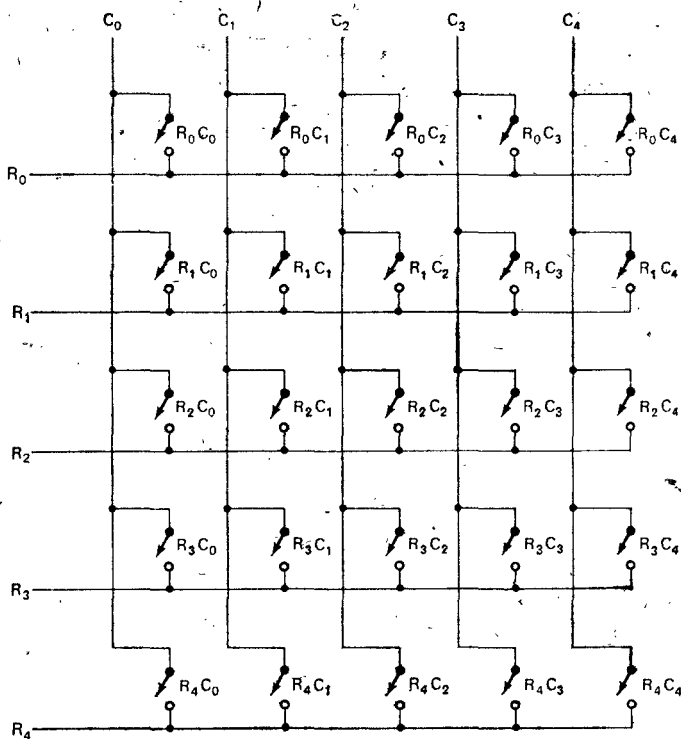


Рис. 4.2. Матрица 5×5 однополюсных нормально разомкнутых ключей.

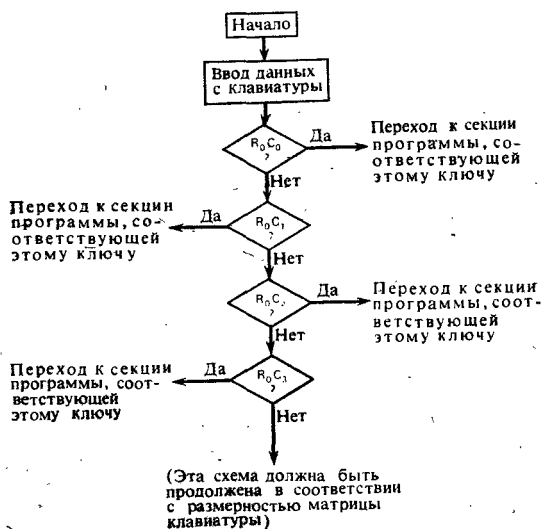


Рис. 4.3. Фрагмент блок-схемы алгоритма определения «координат» замкнутого ключа.

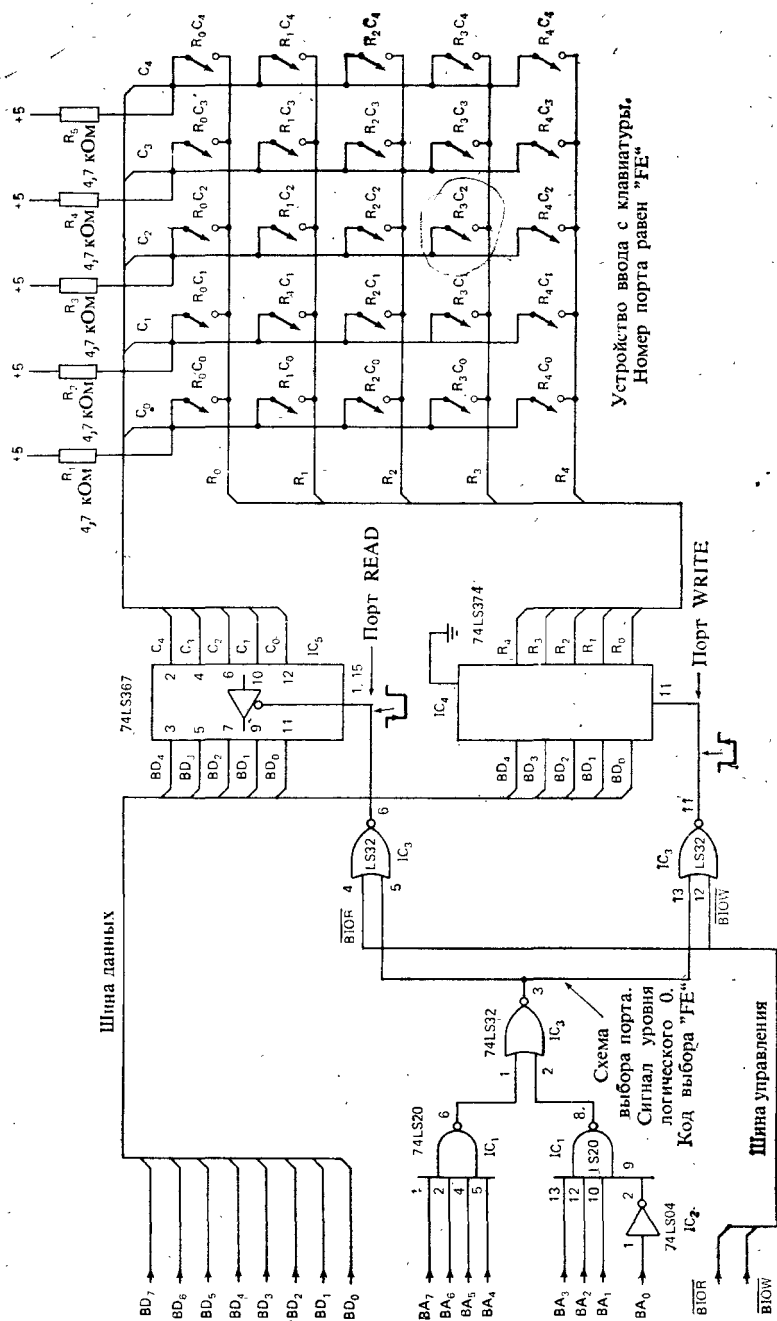


Рис. 4.4. Схема связи матрицы ключей 5×5 (фрагмент) микропроцессорной системы.

Ссылаясь на рис. 4.4, рассмотрим функцию каждого компонента системы ввода-вывода. Сначала обратим внимание на матрицу ключей. Ключи располагаются в матрице следующим образом. Один вывод каждого ключа соединен с одной из вертикальных линий $C_0—C_4$, другой вывод соединен с одной из горизонтальных линий $R_0—R_4$. Горизонтальные линии $R_0—R_4$ являются входными линиями матрицы ключей клавишного пульта. Вертикальные линии являются выходными линиями матрицы ключей. В каждый момент времени только одна из горизонтальных линий $R_0—R_4$ может иметь уровень логического 0, т. е. когда на линии R_0 , например, установлен уровень логического 0, то на линиях R_1, R_2, R_3 и R_4 должен быть уровень логической 1. Таким взаимно исключающим способом выполняется установка уровня, соответствующего логическому 0, для каждой горизонтальной линии. Понятие «взаимно исключающий способ» означает, что в любой конкретный момент времени только одна горизонтальная линия может иметь уровень логического 0.

В то время как единственная горизонтальная линия имеет уровень логического 0, вертикальные выходные линии матрицы клавишного пульта проверяются микропроцессорной системой. Если при этом обнаруживается замкнутый ключ, то соответствующая ему вертикальная линия получает уровень логического 0. Вертикальная выходная линия будет иметь уровень логического 0 тогда и только тогда, когда замкнутый ключ соединяет ее с входной линией, на которой будет установлен уровень логического 0. Чтобы подробнее рассмотреть этот вопрос, допустим, что в данный момент замкнут ключ, имеющий на схеме обозначение R_3C_2 . Такое обозначение показывает, что соответствующий ключ в положении «замкнуто» соединяет входную линию 3 и выходную линию 2. Во время «просмотра» клавиатуры с помощью программных средств системы могут иметь место следующие события. Под термином «просмотр» мы понимаем, что система поочередно переводит каждую входную линию в состояние логического 0 и анализирует состояние выходов матрицы.

Линия R_0 устанавливается в состояние логического 0. Все другие входные линии при этом имеют уровень логической 1. Пока на горизонтальной линии R_0 поддерживается уровень логического 0, выходные линии анализируются с помощью программного обеспечения системы. Ниже будет разъяснено, как горизонтальные линии устанавливаются в состояние логического 0 и каким образом с помощью программного обеспечения проверяются выходные линии. Если ключ R_3C_2 замкнут и на входной линии R_0 установлен уровень логического 0, на всех выходных линиях будет уровень логической 1. Это объясняется тем, что на вертикальных линиях уровень напряжения доведен до

+5 В через согласующие резисторы PR_1 — PR_5 . Поскольку линия R_0 не соединена с помощью ключа ни с одной вертикальной линией, ни на одной выходной линии не может быть уровня напряжения, соответствующего логическому 0.

Аналогично система будет реагировать на установку уровня логического 0 на входных линиях R_0 , R_1 , R_2 и R_4 . Однако, когда на горизонтальной линии R_3 будет установлен уровень логического 0, произойдет следующее. На выходной линии C_2 также установится уровень логического 0. Это объясняется наличием электрического тракта через ключ R_3C_2 . Он образован вертикальной линией C_2 , замкнутым ключом и горизонтальной линией R_3 . После того как система обнаруживает уровень логического 0 на какой-нибудь вертикальной линии, она определяет ключ, замыкание которого обусловило это событие.

Система определяет «координаты» замкнутого ключа путем «электрического» запоминания номера горизонтальной линии, установленной в состояние логического 0, и посредством логического определения номера вертикальной линии, имеющей уровень логического 0. Ниже в этой главе будут рассмотрены существующие программные средства, необходимые для выполнения описанных действий.

4.3. Сигналы горизонтальных линий

Теперь рассмотрим, как на входных линиях по способу взаимного исключения устанавливаются уровни логического 0. Для того чтобы на входных линиях установить уровень логического 0, к клавишному устройству ввода подключается восьмиразрядный фиксатор 74LS374, имеющий на рис. 4.4 обозначение IC_5 . Сигналы на вход этого фиксатора поступают с шины данных. Соответствующие входы обозначены на схеме через BD_0 — BD_4 . Между линиями шины данных и входными линиями матрицы клавишного пульта установлено следующее соответствие: BD_0 — R_0 , BD_1 — R_1 , BD_2 — R_2 , BD_3 — R_3 , BD_4 — R_4 . Микропроцессор формирует слово, логические значения разрядов которого определяют уровни сигналов на входах клавишного пульта, и записывает образованное слово через шину данных в фиксатор 74LS374.

Запись производится, когда система выполняет операцию вывода. Напомним, что при этом она производит следующие действия:

1. Адрес устройства вывода подается на адресную шину.
2. Выводимые данные подаются на шину данных.
3. По шине управления подается сигнал $BIO\bar{W}$.

Рассмотрим, используя схему, представленную на рис. 4.4, как аппаратно реализуется каждое из трех указанных действий. Сначала адрес устройства вывода устанавливается равным

«FE». Схема дешифрирования адреса показана на рис. 4.5, который является фрагментом рис. 4.4. Из рис. 4.5 видно, что схема дешифратора адреса реализуется в виде простого комбинационного логического блока, обеспечивающего дешифрирование 8-разрядного кода, поступающего с шины адреса. Комбинационная схема конструируется для дешифрирования кода 1111 1110 (FE). На выходе схемы 74LS32, показанной на рис. 4.5, уровень

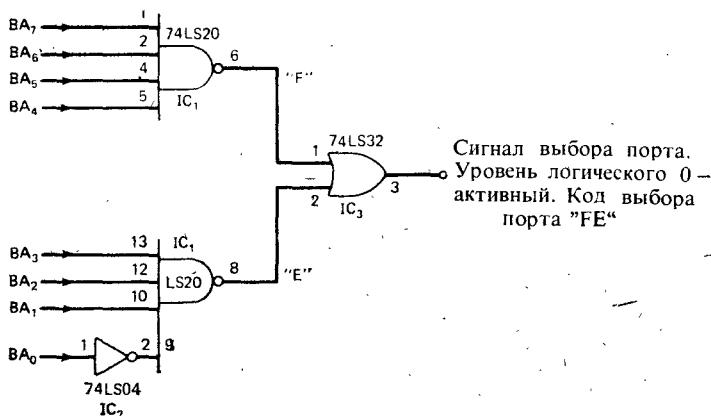


Рис. 4.5. Схема дешифрирования кода выбора порта «FE» (фрагмент схемы, представленной на рис. 4.4).

логического 0 может быть только тогда, когда все восемь разрядов кода на шине адреса совпадают с соответствующими разрядами кода FE. Вывод 3 схемы IC₃ называется линией выбора порта. На этой линии появляется уровень логического 0 каждый раз, как только логическая схема дешифрирования кода выбора порта обнаруживает правильную комбинацию значений двоичных разрядов на восьми младших линиях шины адреса. На этом вопросе стоит сосредоточить внимание, так как он иногда не рассматривается в начальных курсах по микропроцессорным системам. Дело в том, что линия выбора порта может быть активной даже тогда, когда система не связана с устройством ввода.

В подтверждение сказанного рассмотрим следующий пример. Допустим, система читала данные из ячейки памяти с адресом FE. Чтобы активизировать линию выбора порта «FE», шина адреса должна содержать правильную логическую комбинацию. По этой причине события в микропроцессорной системе должны «квалифицироваться» шиной управления. Если же для квалификации событий шина управления не используется, то связаться с выбранными устройствами в определенный момент не будет возможности. Этот факт важно учитывать, когда рассмат-

риваются логические сигналы в системах, управляемых микропроцессором. Сигнал на линии выбора порта может появиться тогда, когда система связана с другими устройствами. Такая ситуация может показаться непонятной, особенно если наблюдение ведется с помощью осциллографа и на его экране обнаруживается, что линия выбора порта активизирована, хотя система, казалось бы, к связи с устройством вывода не готовится.

После дешифрирования кода, поступившего по шине адреса,

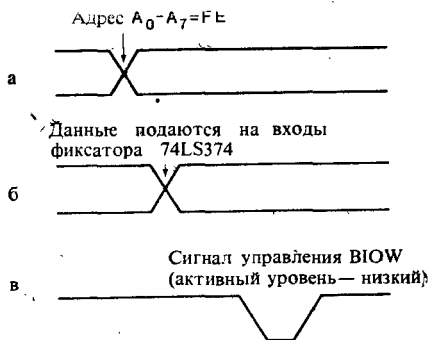


Рис. 4.6. Временная диаграмма сигналов, используемых при записи данных в порт «FE».

уровень логического 0 устанавливается на линии выбора порта и на выводе 13 схемы ИЛИ, обозначенной на рис. 4.4 через IC_3 . Данные по шине данных поступают на входы 3, 4, 7, 8, 13 8-разрядного фиксатора 74LS374. Наконец, по шине управления подается сигнал $BIOR$. Это приводит к установлению на выводе 11 схемы IC_3 уровня логического 0. Затем сигнал $BIOR$ принимает логическое значение 1. На этом этапе данные, имеющиеся на шине данных, запоминаются в 8-разрядном фиксаторе IC_5 типа 74LS374. Соответствующая временная диаграмма представлена на рис. 4.6.

Итак, рассмотрены все фактические события, обусловленные различными логическими значениями сигналов, устанавливаемых микропроцессорной системой на входных линиях клавишного пульта. Кроме того, они аналогичны тем событиям, которые происходят в микропроцессорной системе при обычной записи данных, и совершаются независимо от микропроцессора, используемого в качестве системного контроллера. Форма представления событий может быть различной, но последовательность их наступления должна быть неизменной. Здесь мы не будем учитывать особенности конкретных микропроцессоров, а рассмотрим события, имеющие отношение к микропроцессорным системам с 3 шинами. Такой подход позволит расширить наши общие представления о принципах построения микропроцессорных систем с 3 шинами. Очевидно, что можно спроекти-

ровать ЦП, подходящий для работы в системах с 3 шинами, использование которого облегчит проектирование устройств ввода-вывода.

4.4. Распознавание сигналов на выходах клавишного пульта

Теперь обсудим, каким образом микропроцессор анализирует состояние выходных линий клавишного пульта, показанных на рис. 4.4. Из рис. 4.4 видно, что код выбора порта FE используется в качестве кода выбора при чтении вводимых данных. Он же применялся и для выбора порта записи выходных данных. Таким образом, мы используем этот код как единый код выбора входного и выходного порта. Это вполне допустимо, так как событие, связанное с чтением вводимых данных, никогда не наступает одновременно с записью выводимых данных. Из этого следует, что аппаратные средства для формирования сигнала выбора порта могут быть использованы для двух различных целей, а именно для формирования сигнала выбора порта ввода и сигнала выбора порта вывода. При этом сокращается объем аппаратных средств системы.

Основная идея осуществления операции ввода данных с клавишного пульта состоит в том, что после анализа состояния выходных линий клавишного пульта выполняется операция чтения данных с системного устройства ввода. Система будет выполнять чтение данных, а данные, вводимые в микропроцессор, будут определяться уровнями напряжения на выходных линиях клавишного пульта.

Напомним, что операция чтения данных в системе с 3 шинами выполняется в следующей последовательности:

1. Адрес выбора устройства ввода поступает на адресную шину.

2. Шина данных подготавливается к выполнению операции ввода. (Это означает, что микропроцессор должен перевести входы данных в режим ввода, а не в режим вывода.)

3. Шина управления посылает сигнал $\overline{\text{BIOR}}$.

В рассматриваемом примере адрес порта ввода равен FE. Этот адрес дешифрируется точно таким же способом, как и рассмотренный выше способ дешифрирования адреса при выполнении операции вывода. Это так, поскольку для дешифрирования как кода выбора порта ввода, так и кода выбора порта вывода применяется одно и то же оборудование. Когда обнаруживается код выбора порта, на выводе 5 схемы IC₃ рис. 4.4 появляется уровень логического 0. Теперь по шине управления подается сигнал $\overline{\text{BIOR}}$. Когда сигнал $\overline{\text{BIOR}}$ имеет логическое значение 0 (рабочее состояние), на выводе 6 схемы IC₃ также

будет уровень логического 0. Уровень логического 0 установится и на выводах 1, 15 буфера с тремя состояниями 74LS367, имеющего на схеме обозначение IC₄. Сигнал уровня логического 0 отпирает буфер 74LS367, подключая его выходы к шине данных. Логические значения сигналов на выходах буфера 74LS367 определяются логическими состояниями выходных линий клавишного пульта (рис. 4.7). Микропроцессор будет вводить данные, поступающие на шину данных из буфера 74LS367. Эти данные определяются логическими состояниями выходных

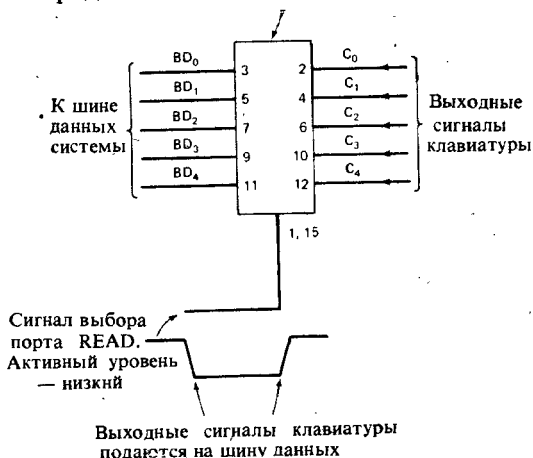


Рис. 4.7. Схема, обеспечивающая считывание данных с выходов клавиатуры (фрагмент схемы, представленной на рис. 4.4).

линий матрицы клавишного пульта. Так данные передаются от устройства ввода к микропроцессору. Теперь на основе полученных данных микропроцессор может «принимать решения».

4.5. Выводы

Мы рассмотрели подключение схем, обеспечивающих работу клавишного пульта, к микропроцессорной системе с 3 шинами, стараясь акцентировать внимание на общих представлениях. Также были представлены современные схемы, с помощью которых могут быть практически реализованы изложенные идеи и методы. Если основные схемы обеспечения ввода-вывода понятны и если ясна идея связи с микропроцессорной системой с 3 шинами, то применение изученных методов при проектировании других схем ввода и вывода окажется простой задачей. Ниже в этой главе обсуждаются методы программного управления портом ввода и вывода.

4.6. Цифровой индикатор

Будем рассматривать индикатор, имеющий поле для отображения 6 шестнадцатеричных цифр. Примером такого индикатора может служить устройство P/N 5082—7340 фирмы Hewlett Packard, выполненное на интегральных схемах. Этот индикатор содержит в себе необходимые схемы дешифрирования. В индикаторе имеется фиксатор, что позволяет подключить его непосредственно к шине данных микропроцессорной системы. Более общие методы фиксации данных в выходном порте системы мы разберем, не используя упомянутый индикатор. Структурная схема подключения индикатора показана на рис. 4.8. Ссылаясь на рисунок, разберем принцип действия этой схемы. Выходной индикатор, имеющий на схеме обозначения DP_1 — DP_6 , получает входные сигналы от 8-разрядных фиксаторов типа 74LS374, обозначенных на схеме через IC_1 — IC_3 . Входы фиксаторов подключаются к линиям BD_0 — BD_7 системной шины данных. Порты вывода, реализованные в виде 8-разрядных фиксаторов, обозначенные на схеме IC_1 , IC_2 и IC_3 , имеют коды выбора порта $F2_{16}$, $F1_{16}$ и $F0_{16}$ соответственно. Когда на адресных линиях BA_4 — BA_7 появляется код 1111, на выходе элемента 7420, т. е. на выводе 6, устанавливается уровень логического 0. Такой же уровень, очевидно, установится и на входе 1 схемы 74LS32 (IC_4). Если при этом на адресной линии BA_3 будет уровень логического 0, этот сигнал откроет схему 74LS42. Если схема 74LS42 открыта, то на выходах 1, 2 и 3 появятся сигналы, значения которых определены двоичными состояниями адресных линий BA_0 — BA_2 . Логические значения сигналов на выходах схемы 74LS42 (IC_5) показаны на рис. 4.9. На этом рисунке также приведены соответствующие коды выбора порта, которые будут формироваться на выходах схемы IC_5 . Кроме того, из рис. 4.9 видно, что когда по адресным линиям поступает код $F0$, на выходе 1 схемы IC_5 появляется уровень логического 0. Очевидно, что и на входе 12 схемы IC_4 установится уровень логического 0. Когда по системной шине управления будет подан сигнал $B10W$, выход 11 схемы IC_4 74LS32 перейдет в состояние логического 0. При переходе вывода 11 от состояния логического 0 к состоянию логической 1 данные, имеющиеся на шине данных, записываются в фиксатор 74LS374 (IC_3).

Аналогично может быть описано функционирование 8-разрядных фиксаторов IC_1 и IC_2 . Запись в 8-разрядный фиксатор IC_1 производится при появлении на адресных линиях кода $F2$, а в фиксатор IC_2 — при появлении кода $F1$. Микропроцессорная система может записывать данные в 8-разрядные фиксаторы одновременно по 8 разрядов. Это означает, что две отображаемые на индикаторе цифры могут быть записаны параллельно. В функции программного обеспечения системы входит регист-

рация тракта, по которому должна обновляться или записываться цифра. Ниже в этой главе будут рассмотрены вопросы программного управления цифровым индикатором.

Выходы устройства 74LS42

вывод 1	вывод 2	вывод 3	Коды адреса
0	1	1	F0 ₁₆
1	0	1	F1 ₁₆
1	1	0	F2 ₁₆

Рис. 4.9. Логические значения сигналов на выводах схемы 74LS42, показанной на рис. 4.8, соответствующие различным кодам адреса.

4.7. Программное управление клавишным пультом

В оставшейся части этой главы рассмотрим программное обеспечение для управления работой клавишного пульта. Из клавишного пульта будет поступать информация о замыкании ключа, которая однозначно соответствует позиции замкнутого ключа и должна быть зарегистрирована. Обсудим используемые на практике экономичные методы разработки и отладки программного обеспечения. Наконец, рассмотрим программное обеспечение, необходимое для функционирования системы, базирующейся на одном из четырех типов микропроцессоров, представленных в этой книге, а именно 8080, 8085, Z80 и 6800.

Системы, основанные на микропроцессорах 8080 и 8085, могут иметь идентичное программное обеспечение. Оно же может использоваться и для систем, основанных на микропроцессоре Z80. Однако мы не приведем здесь программ, написанных с использованием мнемоник микропроцессора Z80. Более важно уметь понимать суть взаимодействия рассматриваемых программных и аппаратных средств микропроцессорных систем управления. Программные средства будем разрабатывать небольшими частями, соответствующими отдельным функциям управления. Потом эти части объединим в функциональную программу программного обеспечения. По сравнению с попыткой представить задачи в виде одной функции этот подход позволяет легче решить общую задачу проектирования программных средств. Причем программное обеспечение в этом случае удобнее отлаживать.

4.8. Программный метод формирования сигналов на входах матрицы клавишного пульта

Рассмотрим программные средства, необходимые для управления аппаратными средствами, выполняющими установку по способу взаимного исключения на входных линиях клавишного пульта уровня, соответствующего логическому 0. Напомним, что только одна входная линия клавишного пульта в любой

момент времени может находиться в состоянии логического 0. Сначала рассмотрим блок-схему, определяющую порядок выполнения указанных действий. Затем представим программное обеспечение, реализующее эту блок-схему, для микропроцессоров 8080, 8085, Z80 и 6800.

Отметим еще раз, что микропроцессоры 8080, 8085 и Z80 могут выполнять программы, разработанные для микропроцессора 8080. Поэтому будем представлять программное обеспечение только в мнемониках микропроцессоров 8080 и 6800. Микропроцессор Z80 может выполнять программы, разработанные для микропроцессора 8080, однако набор команд микропроцессора Z80 шире, чем у микропроцессора 8080. Микропроцессор Z80 имеет свой собственный набор мнемоник. Читателю должно быть понятно, что структура программного обеспечения для разных микропроцессоров может быть различной. Поэтому, используя микропроцессоры в качестве системных контроллеров, следует обращаться к руководствам по их программному обеспечению. Руководства содержат наборы мнемоник конкретных микропроцессоров. Однако общая структура программного обеспечения микропроцессоров подобна структуре программного обеспечения каждого отдельно взятого микропроцессора.

На рис. 4.10 показана блок-схема алгоритма генерации входных сигналов для клавишного пульта. Отметим, что формирование входных сигналов осуществляется с помощью подпрограммы. Начальное значение слова S_1 устанавливается при инициализации микропроцессорной системы. Ниже в этой главе будет точно определено, что подразумевается под термином «инициализация» микропроцессорной системы. Сейчас же достаточно знать, что начальное значение слова S_1 должно быть записано в память перед тем, как микропроцессор начнет работать по программе, блок-схема которой представлена на рис. 4.10. Это требование легко объясняется тем, что, судя по блок-схеме, программа должна принимать решение, зависящее от значения слова S_1 . Значение этого слова должно быть установлено после включения питания, иначе оно будет иметь некоторое произвольное значение, обусловленное случайным состоянием памяти. Случайное значение S_1 приведет к тому, что решение, принимаемое программой после анализа этого слова, окажется ошибочным.

Программа, блок-схема которой дана на рис. 4.10, сначала выбирает слово из памяти. Затем производится проверка на равенство значения этого слова коду 10_{16} . Если факт равенства устанавливается, то это означает, что последняя входная линия уже имела уровень логического 0 и следующей линией, на которой должен устанавливаться уровень логического 0, является линия R_0 . Если текущее значение слова S_1 не равно 10_{16} , то программа просто сдвигает содержимое слова на один разряд

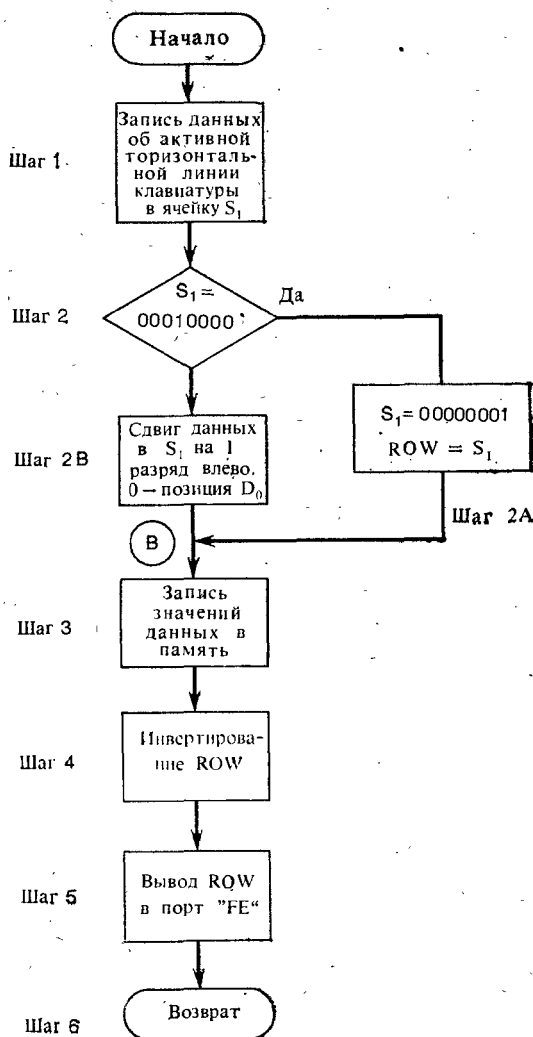


Рис. 4.10. Блок-схема программы генерации входных сигналов матрицы клавишного пульта.

влево. При выполнении этого сдвига в первой позиции слова должно появиться значение 0. Подчеркиваем это, так как различные микропроцессоры выполняют операцию сдвига слова различными способами. Некоторые микропроцессоры могут в первый разряд сдвинутого слова поместить 1. Чтобы точно определить, как выполняется сдвиг влево, необходимо обратиться к руководству по программному обеспечению используемого микропроцессора.

Если анализируемое слово оказывается равным 10_{16} , то согласно блок-схеме программы (см. рис. 4.10) будет установлено новое значение слова, равное 01_{16} . После этого выполнение программы продолжается с точки В, указанной на блок-схеме. На этом шаге программа записывает новое слово в память. По-



Рис. 4.11. Этот рисунок поясняет действия, выполняемые в шагах 3—5 блок-схемы, показанной на рис. 4.10. Слово состояний входных линий матрицы записывается в память. Затем оно инвертируется и записывается в порт вывода. Активной линии соответствует логическое значение 0.

мещенное в память слово будет анализироваться при следующем обращении к этой программе.

После записи нового значения слова в память слово, определяющее значения сигналов на горизонтальных линиях матрицы клавишного пульта, инвертируется. Это действие заключается в установке значения 1 во всех разрядах, за исключением разряда, соответствующего входной линии, подлежащей активизации. Разряд, определяющий активизируемую линию, принимает значение 0. Описанные действия иллюстрируются рис. 4.11. В конце концов, программа выведет сформированное слово в определенный порт вывода.

Напомним, что на рис. 4.4 порт вывода, связанный со входными линиями клавишного пульта, адресовался кодом FE. На рис. 4.12 представлен текст программы для микропроцессора 8080, соответствующий блок-схеме, приведенной на рис. 4.10. Следует учитывать, что эта программа представляет собой один из возможных вариантов реализации блок-схемы, представленной на рис. 4.10. Совершенно ясно, что существует много различных способов программной реализации данной блок-схемы. Это аналогично большому разнообразию методов схемной реализации каких-либо функций.

Какой бы вариант реализации программы ни был избран, программа всегда должна быть хорошо документирована. Ког-

да наступает время отладки программы или программное обеспечение кому-либо передается, хорошая документация будет бесценной. В некоторых случаях стремление получить хорошую

```

311 00EB *****
312 00EB *
313 00EB * ПРОГРАММА ВЫВОДА АКТИВНОГО НАБОРА НА ПУЛЬТ
314 00EB *
315 00EB * ИМЯ ЭТОЙ ПРОГРАММЫ-OROW
316 00EB *
317 00EB * ЭТО РИС.4.12
318 00EB *
319 00EB *****
320 00EB *****
321 00EB *
322 00EB * РИС.4.12-ПОДПРОГРАММА К РИС.4.10 В МНЕМОНИКЕ 8080
323 00EB *
324 00EB *****
325 00EB *
326 00EB *
327 00EB 3A 04 10 OROW LDA KROW ПОВТОРНЫЙ ВЫЗОВ АКТИВНОГО НАБОРА
328 00EB *
329 00EB ***** (ШАГ 2)
330 00EB *
331 00EB FE 10 CPI 10H СКАНИРОВАНИЕ НАБОРА ЗАКОНЧЕНО???
332 00EB C2 F8 00 JNZ OROW1 ЕСЛИ НЕТ,ТО ШАГ 2B
333 00F3 *
334 00F3 ***** (ШАГ 2A)
335 00F3 *
336 00F3 3E 01 MVI A,01H ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАБОР
337 00F5 C3 F9 00 JMP ST3 ИДТИ НА ШАГ 3
338 00F8 *
339 00F8 ***** (ШАГ 2B)
340 00F8 *
341 00F8 07 OROW1 RLC СДВИНУТЬ ДАННЫЕ ВЛЕВО,0 В DO
342 00F9 *
343 00F9 ***** (ШАГ 3)
344 00F9 *
345 00F9 32 04 10 ST3 STA KROW СОХРАНИТЬ АКТИВНЫЙ НАБОР
346 00FC *
347 00FC ***** (ШАГ 4)
348 00FC *
349 00FC 2F CMA ИНВЕРТИРОВАТЬ СЛОВО НАБОРА
350 00FD *
351 00FD ***** (ШАГ 5)
352 00FD *
353 00FD D3 FE OUT OFEH ВЫВЕСТИ АКТИВНЫЙ НАБОР В ПОРТ FE
354 00FF *
355 00FF ***** (ШАГ 6)
356 00FF *
357 00FF C9 RET
358 0100 *
359 0100 *

```

Рис. 4.12. Программа на языке ассемблера микропроцессора 8080, соответствующая блок-схеме, представленной на рис. 4.10.

документацию стимулирует использование простых приемов программирования и оправдывает прямолинейность в способах программной реализации блок-схем.

На рис. 4.13 приведена программа, соответствующая блок-схеме, показанной на рис. 4.10; данная программа предназначена для микропроцессора 6800. В микропроцессоре 6800 при-

LF413 T=00003 IS ON CRO0020 USING 00009 BLKS R=1410

```

0001 1 PAGE 1 F412 TUE OCT 02,1979 02:27:04.13
0002
0003
0004
0005 00001 NAM F412
0006 00002 *
0007 00003 *
0008 00004 * ЭТА ПРОГРАММА-ВЕРСИЯ РИС.4.12 ДЛЯ 6800
0009 00005 *
0010 00006 0000 ORG S00
0011 00007 *
0012 00008 ***** (ШАГ 2)
0013 00009 *
0014 00010 0000 81 10 CMP A #S10 СКАНИРОВАНИЕ НАБОРА ЗАКОНЧЕНО??
0015 00011 0002 26 05 BNE ST2B ЕСЛИ НЕТ, ТО ШАГ 2В
0016 00012 *
0017 00013 ***** (ШАГ 2А)
0018 00014 *
0019 00015 0004 86 01 LDA A #S01 ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАБОР
0020 00016 0006 7E 000A JMP ST3 ИДТИ НА ШАГ 3
0021 00017 *
0022 00018 ***** (ШАГ 2В)
0023 00019 *
0024 00020 0009 48 ST2B ASL A СДВИНУТЬ ВЛЕВО, 0 В DO
0025 00021 *
0026 00022 ***** (ШАГ 3)
0027 00023 *
0028 00024 000A B7 1000 ST3 STA A KROW
0029 00025 *
0030 00026 ***** (ШАГ 4)
0031 00027 *
0032 00028 000D 43 COM A ИНВЕРТИРОВАТЬ СЛОВО НАБОРА
0033 00029 *
0034 00030 ***** (ШАГ 5)
0035 00031 *
0036 00032 000E 01 FE STA A #S00FE ВЫВЕСТИ АКТИВ. НАБОР В ПОРТ FE
0037 00033 *
0038 00034 ***** (ШАГ 6)
0039 00035 *
0040 00036 0010 39 RST ВЫЙТИ ИЗ ПОДПРОГРАММЫ
0041 00037 *
0042 00038 *
0043 00039 ***** КОНЕЦ ЭТОЙ ПРОГРАММЫ *****
0044 00040 *
0045 00041 * РАЗМЕСТИТЬ ПЕРЕМЕННЫЕ В ПАМЯТИ
0046 00042 *
0047 00043 1000 KROW EQU S1000
0048 00044 *
0049 00045 END ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА
0050 1 PAGE 2 F412 TUE OCT 02,1979 02:27:04.13
0051
0052
0053
0054 KROW 1000 00043 00024
0055 ST2B 0009 00020 00011
0056 ST3 000A 00024 00016

```

Рис. 4.13. Программа на языке ассемблера микропроцессора 6800, соответствующая блок-схеме, представленной на рис. 4.10.

меняется способ обращения к устройствам ввода-вывода, аналогичный способу обращения к памяти. Микропроцессор 6800 не имеет команды типа OUT для записи данных во внешние устройства, которые условно считаются ячейками памяти.

4.9. Опрос выходных линий клавишного пульта с помощью программных средств

Теперь рассмотрим метод программного управления аппаратными средствами с целью точного определения активной выходной линии среди линий C_0, C_1, C_2, C_3, C_4 . Блок-схема алгоритма

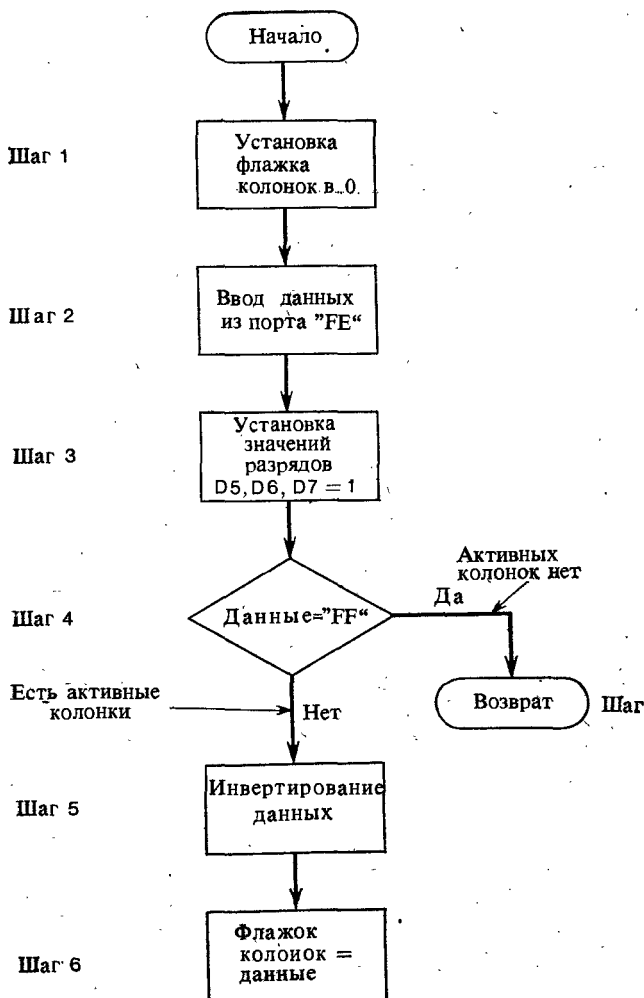


Рис. 4.14. Блок-схема программы определения активных выходов матрицы клавиатуры.

определения активной выходной линии показана на рис. 4.14. Сначала программно «флажку выходов» присваивается нулевое значение. Этот флажок будет использоваться вызывающей программой для определения наличия активных выходных ли-

```

278 OODB *****
279 OODB *
280 OODB *   ЭТА ПОДПРОГРАММА-РИС.4.15
281 OODB *
282 OODB *****
283 OODB *
284 OODB ***** (ШАГ 1)
285 OODB *
286 OODB AF COLM XRA A
287 OODC 32 01 10 STA CFLAG   УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА=0
288 OODF *
289 OODF ***** (ШАГ 2)
290 OODF *
291 OODF DB FE IN OFEH   ВХОДНЫЕ ДАННЫЕ ИЗ ПОРТА FE
292 OOE1 *
293 OOE1 ***** (ШАГ 3)
294 OOE1 *
295 OOE1 F6 EO ORI OEON   УСТАНОВИТЬ ВИТЫ D5,D6,D7=1
296 OOE3 *
297 OOE3 ***** (ШАГ 4)
298 OOE3 *
299 OOE3 FE FF CFI OFEH

300 OOE5 C8 RZ   ШАГ (4A)
301 OOE6 *
302 OOE6 ***** (ШАГ 5)
303 OOE6 *
304 OOE6 2F CMA   ИНВЕРТИРОВАТЬ ДАННЫЕ
305 OOE7 *
306 OOE7 ***** (ШАГ 6)
307 OOE7 *
308 OOE7 32 01 10 STA CFLAG   УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
309 OOE6 C9 RET   ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
310 OOE6 *

```

Рис. 4.15. Программа на языке ассемблера микропроцессора 8080, соответствующая блок-схеме, представленной на рис. 4.14.

ний. Затем вводятся данные из порта ввода клавиатуры. На третьем шаге в три старших разряда введенного слова записываются значения 1. Последнее действие выполняется потому, что входной порт, из которого были прочитаны данные, соединен только с пятью младшими разрядами шины данных. Чтобы неопределенность данных в старших трех разрядах введенного слова не повлияла на выполнение следующих шагов программы, целесообразно «маскировать» указанные разряды.

На четвертом шаге данные, полученные из порта ввода клавишного пульта, анализируются. Если во всех разрядах обнаруживаются единицы, т. е. код FF, то среди выходных линий

LF416 T=00003 IS ON CRO0020 USING 00009 BLKS R=1410

```

0001 1 PAGE 1 F4 TUE OCT 02,1979 01:59:27.45
0002
0003
0004
0005 00001 NAM F4.15
0006 00002 *
0007 00003 *
0008 00004 * ЭТО ВЕРСИЯ РИС.4.15 ДЛЯ 6800
0009 00005 *
0010 00006 *
0011 00007 0000 JRG 00 УСТАНОВИТЬ НАЧ.АДРЕС ЭТОЙ ПРОГ.
0012 00008 *
0013 00009 * УСТАНОВИТЬ ПОЛЕ ПЕРЕМЕННЫХ
0014 00010 *
0015 00011 1001 CFLAG EQU S1001
0016 00012 *
0017 00013 * НАЧАЛО ПРОГРАММЫ
0018 00014 *
0019 00015 *
0020 00016 * (ШАГ 1)
0021 00017 *
0022 00018 0000 86 00 COLM LDA A #S00
0023 00019 0002 97 00 STA A COLM УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
0024 00020 *
0025 00021 * (ШАГ 2)
0026 00022 *
0027 00023 0004 96 FE LDA A S00FE ВХОДНЫЕ ДАННЫЕ ИЗ ПОРТА FE
0028 00024 *
0029 00025 ***** (ШАГ 3)
0030 00026 *
0031 00027 0006 8A E0 ORA A #S00 УСТАНОВИТЬ БИТЫ D5,D6,D7=1
0032 00028 *
0033 00029 ***** (ШАГ 4)
0034 00030 *
0035 00031 0008 81 FF CMP A #S00FF
0036 00032 000A 27 01 BEQ NEXT ЕСЛИ НОЛЬ, ТО ПЕРЕХОД
0037 00033 000C 39 RTS ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГ. ЕСЛИ НЕ 0
0038 00034 *
0039 00035 ***** (ШАГ 5)
0040 00036 *
0041 00037 000D 43 NEXT COM A ИНВЕРТИРОВАТЬ ДАННЫЕ
0042 00038 *
0043 00039 ***** (ШАГ 6)
0044 00040 *
0045 00041 000E B7 1001 STA A CFLAG УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
0046 00042 0011 39 RTS ВЫЙТИ ИЗ ПОДПРОГРАММЫ
0047 00043 *
0048 00044 *
0049 00045 * КОНЕЦ ПОДПРОГРАММЫ
0050 00046 *
0051 00047 END ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА
0052 1 PAGE 2 F4 TUE OCT 02,1979 01:59:27.45
0053
0054
0055
0056 CFLAG 1001 00011 00041
0057 COLM 0000 00018 00019
0058 NEXT 000D 00037 00032

```

Рис. 4.16. Программа на языке ассемблера микропроцессора 6800, соответствующая блок-схеме, представленной на рис. 4.14.

клавиатуры нет активной (она должна находиться в состоянии логического 0). Это условие будет выполняться в тех случаях, когда активизируемые входные линии не соединены с какой-либо выходной линией матрицы клавиатуры. Если условие выполняется, то происходит выход из подпрограммы, а флажок выходов сохраняет нулевые значения своих разрядов. Это означает, что активных выходов нет.

Если обнаруживается наличие активной выходной линии, то данные, полученные из порта ввода, инвертируются. Теперь флажок выходов приобретает значение, равное значению слова, полученного после выполнения инвертирования на 5-м шаге алгоритма. Наконец, происходит возвращение к вызвавшей программе. Согласно рассмотренной блок-схеме написаны программы для микропроцессора 8080 (рис. 4.15) и для микропроцессора 6800 (рис. 4.16).

4.10. Вычисление весового значения ключа

Каждому ключу в матрице клавиатуры ставится в соответствие число из диапазона $0-N$; так, первому ключу, обозначенному через R_0C_0 , соответствует число 0, а последнему ключу R_4C_4 — число N . На рис. 4.17 показаны веса, соответствующие ключам матрицы клавиатуры. Функцию ключа на этой стадии определять не будем. Вес, приданный каждому ключу, позволяет вычислить на основании номера строки и номера столбца матрицы клавиатуры, какой ключ был замкнут. На рис. 4.18 представлена блок-схема алгоритма вычисления веса определенного замкнутого ключа. Рассматривая блок-схему, будем подразумевать, что информация о номере активного столбца хранится в памяти по адресу, обозначенному нами «флажок выходов», а информация о номере строки матрицы, на пересечении которой с активным столбцом находится замкнутый ключ, находится в памяти по адресу, обозначенному как «входное слово».

Ссылаясь на блок-схему, подробно рассмотрим, как действует эта программа. Сначала из памяти выбирается значение «флажка выходов», содержащего информацию о номере искомого столбца. «Флажок выходов» может иметь одно из следующих пяти двоичных значений:

00000001 (0-й столбец);

00000010 (1-й столбец);

00000100 (2-й столбец);

00001000 (3-й столбец);

00010000 (4-й столбец).

Затем, как видно из рис. 4.18, счетчик T_1 принимает значение 0. Потом счетчик T_1 станет равным значению 0, 1, 2, 3 или 4, соответствующему номеру активного выхода клавиатуры. Для выполнения этого программа будет сдвигать «флажок состоя-

ний» каждый раз на один разряд вправо. После первого сдвига в позиции D_7 появится значение 0 (рис. 4.19). Затем сдвинутое слово проверяется на равенство нулю. Если все разряды слова равны 0, то активных выходов нет. В этом случае значение счетчика T_1 равно 0.

Если значение сдвинутого слова данных при проверке на шаге 4 (рис. 4.18) окажется не равным 0, то значение счетчика

	C_0	C_1	C_2	C_3	C_4
R_0	X 0	X 1	X 2	X 3	X 4
R_1	X 5	X 6	X 7	X 8	X 9
R_2	X 10	X 11	X 12	X 13	X 14
R_3	X 15	X 16	X 17	X 18	X 19
R_4	X 20	X 21	X 22	X 23	X 24

Рис. 4.17. Десятичные значения весов, соответствующие позициям в матрице 5×5 клавишного пульта.

T_1 увеличивается на 1. Сдвиги и проверка слова «флажок выходов», сопровождаемые увеличением значения счетчика на 1, продолжаются до тех пор, пока сдвигаемое слово данных не станет равным 0. На этой стадии значение счетчика T_1 временно записывается в память.

Теперь, согласно блок-схеме, изображенной на рис. 4.18, подобные же действия выполняются над значением «входного слова». В данном случае используется счетчик T_2 .

Наконец, после того как в память по адресам T_2 и T_1 записаны соответственно десятичные значения номера строки и номера столбца, вес замкнутого ключа вычисляется по формуле:

$$\text{Вес ключа} = (5 \times \text{номер столбца}) + \text{Номер строки}.$$

Вычисленный вес однозначно определяет, какой ключ был замкнут. Теперь может быть выполнен переход к программной секции, которая соответствует конкретной функции ключа.

Однако, прежде чем выполнить указанный переход, следует убедиться в том, что ключ, которому соответствует вычисленное значение веса ключа матрицы клавиатуры, действительно был замкнут. А также до тех пор, пока ключ не будет разомкнут,

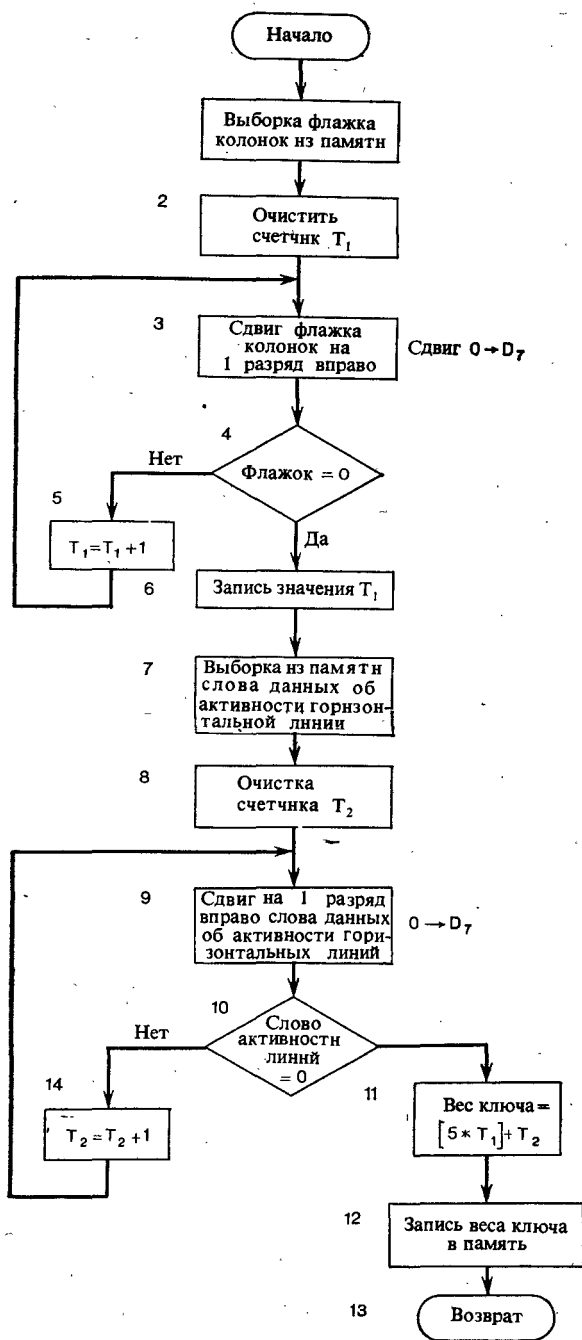


Рис. 4.18. Блок-схема программы вычисления десятичного значения веса нажатой клавиши.

никаких действий выполнять нельзя. В рассматриваемой системе используются следующие критерии определения факта действительного замыкания и факта действительного размыкания ключа:

Ключ считается действительно замкнутым, если микропроцессорная система вычислит один и тот же вес ключа 50 раз подряд.

Ключ считается действительно разомкнутым, если микропроцессорная система обнаружит его разомкнутым в течение

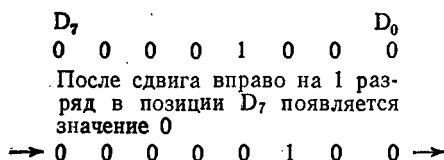


Рис. 4.19. Сдвиг слова данных вправо на один разряд. В позиции D_7 появляется логическое значение 0.

50 последовательных просмотров входной горизонтальной линии матрицы клавиатуры.

Отметим, что выбор числа последовательных подтверждений факта нахождения ключа в определенном состоянии является субъективным. Благодаря последовательным подтверждениям устраняется влияние эффекта механической вибрации клавиши, существующего в клавишных устройствах. Интервал времени, в течение которого ключ находится в неопределенном состоянии, может быть различным для различных клавиатур.

4.11. Программные средства учета эффекта вибрации клавиатуры

Теперь рассмотрим программное обеспечение, используемое для учета эффекта вибрации клавиш. Разберем блок-схему, отражающую последовательность происходящих в системе событий. Потом представим конкретную программу, необходимую для реализации этой блок-схемы. В задаче можно выделить две части. Первая заключается в разработке программы, обеспечивающей 50-кратное последовательное подтверждение факта обнаружения одного ключа в замкнутом состоянии. Вторая часть задачи состоит в установлении того, что в течение 50-кратного последовательного сканирования входных линий матрицы клавиатуры не обнаруживается активной выходной линии клавиатуры. Эти частные задачи соответствуют вопросам определения фактов действительного нажатия и освобождения определенной клавиши.

На рис. 4.20 представлена блок-схема, определяющая последовательность действий, выполняемых системой при вводе информации о нажатии одной клавиши. Отметим, что эта программа использует рассмотренную выше подпрограмму, блок-схема которой дана на рис. 4.10; эта подпрограмма использует



Рис. 4.20. Блок-схема программы ввода информации в микропроцессорную систему о нажатии одной клавиши.

ется для определения номера активизируемой входной линии матрицы клавиатуры. Кроме того, в данной программе используется подпрограмма обнаружения активной выходной линии клавиатуры. Если клавиша не нажата, то «флажок выходов» имеет значение 0. В этом случае управление передается (рис. 4.20) подпрограмме активизации входных линий, которая установит на следующей входной линии уровень логического 0. Этот цикл выполняется системой до тех пор, пока пользователь не нажмет на клавишном пульте какую-либо клавишу.

Если на шаге 3 (рис. 4.20) обнаруживается значение «флажка выходов», отличное от нуля, то данная программа вызовет подпрограмму вычисления веса ключа, принимающего значения

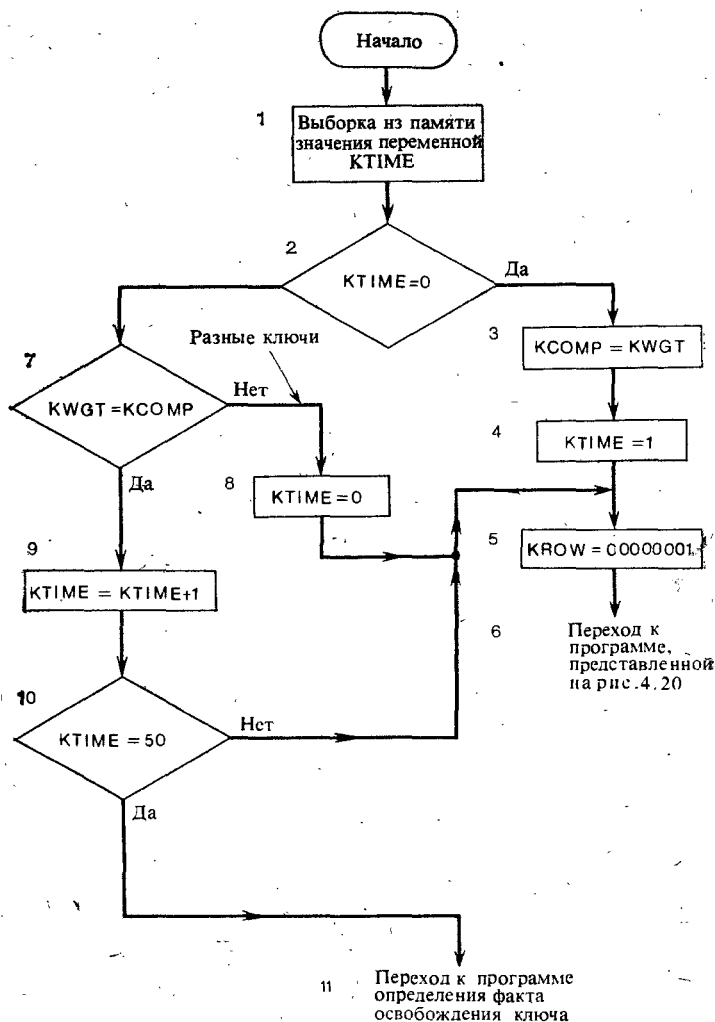


Рис. 4.21. Блок-схема программы определения 50 последовательных подтверждений нажатия определенной клавиши.

в диапазоне от 0 до 24. Программа записывает это число в память по адресу, обозначенному как **KWGT**.

Итак, допустим, система один раз обнаружила активный выход клавиатуры. Теперь должна быть выполнена 50-кратная проверка активности одной и той же выходной линии клавиатуры, определяемой замыканием одного ключа. Блок-схема соответствующего алгоритма показана на рис. 4.21. На данной

блок-схеме счетчик KTIME используется для регистрации числа последовательных поступлений в систему сигналов, соответствующих замыканию одного и того же ключа, и для проверки достижения этого числа заданному количеству подтверждений.

Если счетчик KTIME при проверке на шаге 2' (см. рис. 4.21) равен 0, то значит, что от определенного ключа в систему поступил первый сигнал. Так как сигнал поступил в первый раз, необходимо запомнить соответствующее весовое значение ключа. Это делается посредством присваивания переменной KCOMP весового значения ключа, сигнал от которого поступил в систему. Весовое значение этого ключа всегда запоминается в памяти по адресу KWGT. После присваивания переменной KCOMP на 3-м шаге нового значения устанавливается значение счетчика KTIME равным 1 и происходит переход к секции программ, обеспечивающей прием других сигналов от клавиатуры. Прежде чем перейти к этой секции программы, система восстанавливает значение счетчика активных входных линий клавиатуры KROW. Устанавливаемое начальное значение определяет, что теперь уровень логического 0 должен подаваться на первую горизонтальную линию матрицы клавиатуры. Это действие выполняется на 5-м шаге рассматриваемого алгоритма. Когда информация о замыкании ключа вводится в систему во второй раз, значение счетчика KTIME не будет равно 0. Поэтому после шага 2 выполняется шаг 7. На 7-м шаге сравниваются значения переменных KWGT и KCOMP. Если сравниваемые значения не равны, то выполняется, согласно блок-схеме, переход к шагу 8. На 8-м шаге счетчику KTIME присваивается значение 0, и система будет снова пытаться получить 50 последовательных подтверждений факта нажатия определенной клавиши. Если значения переменных KWGT и KCOMP равны, то после выполнения шага 7 будет выполняться шаг 9. На шаге 9 значение счетчика KTIME увеличивается на 1. Тем самым учитывается еще одно последовательное поступление информации о замыкании одного и того же ключа. Потом система проверяет значение счетчика KTIME. Эта проверка выполняется на 10-м шаге алгоритма. На 10-м шаге фактически проверяется, было ли 50 раз подряд зафиксировано одно и то же значение веса ключа.

Если значение счетчика KTIME не равно 50, то система переходит к выполнению шага 5, назначение которого было описано выше. Затем система снова возвращается к программной секции, обеспечивающей ввод информации о состояниях выходных линий клавишного пульта.

Если на шаге 10 значение счетчика KTIME равно 50, то система выполняет следующие действия. Сначала отмечается, что было зафиксировано действительное нажатие клавиши. Соответствующее весовое значение ключа запоминается в двух ячейках памяти с адресами KWGT и KCOMP. Система использует

ячейки KWGT для сохранения веса действительно замкнутого ключа.

После установления факта действительного нажатия клавиши система перейдет к программной секции, которая обеспечит определение факта отжатия клавиши по принципу 50-кратного подтверждения. До тех пор пока ключ остается в замкнутом состоянии, функция, определяемая им, выполняться не должна. Отметим, что микропроцессорная система должна обладать таким быстродействием, что при оценке общей производительности системы временем, которое тратится на подтверждение нахождения клавиши в том или ином состоянии, можно пренебречь. А время выполнения функции соответствующей клавиши должно быть много меньше, чем время, которое эта клавиша находится в нажатом состоянии.

Если при полном сканировании входных линий матрицы клавиатуры не обнаруживается ни одной активной выходной линии, то фиксируется отсутствие нажатой клавиши. Блок-схема для реализации этой функции показана на рис. 4.22. Сначала переменной KROW присваивается такое значение, которое обеспечит сканирование входных линий клавиатуры, начиная с первой линии. Затем счетчику KTIME присваивается значение 0. В этой программной секции счетчик KTIME используется для подсчета количества полных сканирований клавиатуры, во время которых не обнаруживалось замкнутого ключа. Теперь система вызывает подпрограмму с именем ROW (шаг 3 на блок-схеме). Эта подпрограмма готовит и выдает данные, определяющие значения уровней на входных линиях клавиатуры. После этого система обращается к подпрограмме COL.

Подпрограмма COL проверяет состояние выходных линий клавиатуры. Если активных выходных линий нет, то подпрограмма COL присваивает «флажку выходов» значение 0. Если же активные выходные линии обнаруживаются, то управление передается в начальную точку программы, что показано на блок-схеме стрелкой, идущей от шага 5 к шагу 1.

Если активных выходных линий не обнаружено, то «флажок выходов» имеет значение 0 и система перейдет после шага 5 к шагу 9. На 9-м шаге значение счетчика NROW увеличивается на 1. Счетчик NROW используется для регистрации количества последовательных выполнений подпрограммы COL, во время которых система не обнаруживает активных выходных линий. Счетчик NROW фактически указывает, какая из горизонтальных линий матрицы в настоящее время активизирована. Когда значение NROW равно 5, пятая по счету горизонтальная линия матрицы была активизирована. Однако активных выходных линий система не обнаружила.

В этом случае система, выполняя программу, представленную блок-схемой на рис. 4.22, перейдет от шага 10 к шагу 6.

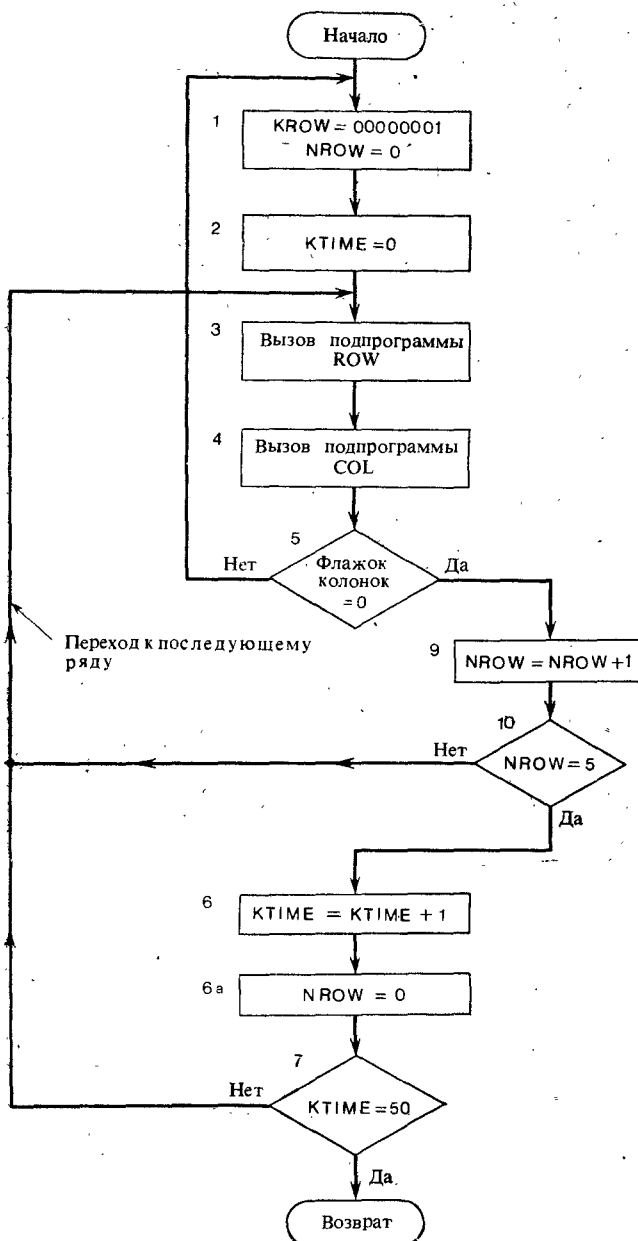


Рис. 4.22. Блок-схема программы определения факта освобождения клавиши клавиатуры.

На шаге 6 счетчик KTIME увеличивается на 1. Этот счетчик регистрирует количество полных сканирований матрицы клавиатуры, в течение которых замыканий ключа не было обнаружено.

Как только значение счетчика KTIME станет равным 50, завершится выполнение полной программы, обеспечивающей ввод и обработку информации в связи с нажатием одной клавиши. Видно, что при этом в системе происходит много собы-

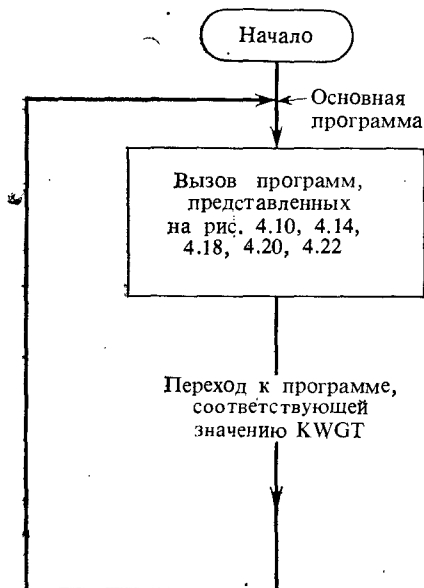


Рис. 4.23. Блок-схема основной программы, вызывающей подпрограмму ввода значения ключа.

тий. Когда система переходит от программы KREL, предназначенной для определения факта освобождения клавиши, к основной программе, выполняется программная проверка значения введенного веса ключа и указывается точка входа в соответствующую системную программу. Эта системная программа используется при реализации функции, определяемой нажатой клавишей.

На рис. 4.23 дана общая блок-схема основной программы, в которой используется подпрограмма KEYIN. Возвращение из подпрограммы KEYIN к основной программе осуществляется только после установления ею факта нажатия и последующего освобождения одной клавиши. То, что клавиша была действительно нажата, проверяется 50 раз подряд, ее освобождение констатируется также после 50-кратного подтверждения. Поэтому подпрограмма KEYIN не возвращает управление до тех пор, пока после нажатия клавиши не наступит ее действительное освобождение.

Затем основная программа, основываясь на значении переменной KWGT, установленном подпрограммой KEYIN, определяет адрес перехода к программной секции, выполнение которой реализует функцию замкнутого ключа. Мы видим, что принципы программного управления клавиатурой обеспечивают универсальность ее применения. Функции клавиш легко изменяются посредством изменения программного обеспечения.

В нашей системе каждой клавише было приписано соответствующее значение веса KWGT, для отображения которого на индикаторе используется два младших разряда. Это значит, что при нажатии клавиши, имеющей в матрице ключей обозначение R₄C₄, на индикаторе появятся цифры 000024, соответствующие весу нажатой клавиши. С помощью этой программы можно убедиться в правильности работы подпрограммы KEYIN. После этого следует назначить клавишам фактические функции.

4.12. Инициализация программы

Перед тем как начать обсуждение программы выполнения общих системных функций, разберем вопрос инициализации системного программного обеспечения. Обычно эта программная секция выполняется только при включении питания системы или когда оператор нажмет кнопку начальной установки системы. Установка начального состояния микропроцессорных систем рассмотрена в гл. 2.

В секции инициализации программы определяются начальные значения переменных, которые в дальнейшем потребуются при выполнении программы. Например, устанавливается начальное значение указателя стека и маска прерываний. Мы еще не обсуждали вопросы обработки прерываний. Этому посвящена гл. 6.

Для нашей программы необходима начальная установка значений следующих переменных:

1. Указатель стека.
2. KROW=00000001.
3. NROW=0.
4. KTIME=0.
5. COLFG=0 (флажок выходов).
6. Индикатор=FF0000.

Выполнение пункта 6 обеспечит вывод на индикатор заранее известного значения.

Программа для выполнения всех функций системы приведена на рис. 4.24. На этом рисунке представлена программа для микропроцессора 8080, а на рис. 4.25 дана программа для микропроцессора 6800.

```

1 0000
2 0000
3 0000
4 0000
5 0000
6 0000
7 0000
8 0000
9 0000
10 0000
11 0000
12 0000 31 FF 13 LXI SP,13FFH ИНИЦИАЛИЗИРОВАТЬ УКАЗАТЕЛЬ СТЕКА
13 0003 AF A XRA A НАЧ.ЗНАЧ. ФЛАЖКОВ,ЗВНУЛ. АККУМУЛЯТОРА
14 0004 FB EI ВОЗМОЖНЫ ПРЕРЫВАНИЯ
15 0005 D3 F0 OUT OF0H ВЫВЕСТИ НОЛЬ НА ДИСПЛЕЙ
16 0007 D3 F1 OUT OF1H
17 0009 D3 F2 OUT OF2H УСТАНОВИТЬ ДИСПЛЕЙ=000000
18 000B 32 00 10 STA KTIME KTIME=0
19 000E 32 01 10 STA CFLAG CFLAG=0
20 0011 32 02 10 STA KWGHT KWGHT=0
21 0014 32 03 10 STA KCOMP KCOMP=0
22 0017 3C INR A АККУМУЛЯТОР=1
23 0018 32 04 10 STA KROW KROW=00000001
24 001B *
25 001B *
26 001B * ТЕПЕРЬ УСТАНОВИМ АДРЕСА ПЕРЕМЕННЫХ
27 001B *
28 001B KTIME EQU 1000H
29 001B CFLAG EQU KTIME+1
30 001B KWGHT EQU CFLAG+1
31 001B KCOMP EQU KWGHT+1
32 001B KROW EQU KCOMP+1
33 001B NROW EQU KROW+1
34 001B *
35 001B *
36 001B ***** ДАЛЕЕ НАЧИНАЕТСЯ ПРОГРАММА *****
37 001B *
38 001B * НАЧАЛО РИСУНКА 4.20
39 001B *
40 001B CD EB 00 SROW CALL OROW ВЫВЕСТИ АКТИВНУЮ СТРОКУ
41 001E CD DB 00 CALL COLM ВЫЗВАТЬ ПОДПРОГ. ОБР. СТОЛБЦА
42 0021 3A 01 10 LDA CFLAG НЕОБХОДИМО ПРОВЕРИТЬ ФЛАЖОК СТОЛБЦА
43 0024 FE 00 CPI 00H ПРОВЕРИТЬ СТОЛБЕЦ НА АКТИВНОСТЬ
44 0026 CA 1B 00 JZ SROW СТОЛБЕЦ НЕ АКТИВЕН,ПРОВЕРИТЬ ЦИКЛ
45 0029 CD B1 00 CALL KEYW СТОЛБЕЦ АКТИВЕН,КАКОЙ ИМЕННО???
46 002C *
47 002C * КОНЕЦ РИСУНКА 4.20
48 002C *
49 002C * НАЧАЛО РИСУНКА 4.21
50 002C *
51 002C ***** ИМЕЕМ АКТИВНУЮ КЛАВИШУ *****
52 002C *
53 002C ***** (ШАГ 1)
54 002C *
55 002C 3A 00 10 LDA KTIME ПОВТОРНЫЙ ВЫЗОВ KTIME ИЗ ПАМЯТИ
56 002F *
57 002F *
58 002F ***** (ШАГ 2)
59 002F *
60 002F FE 00 CPI 00H KTIME=0??

```

Рис. 4.24. Программа ввода информации с клавиатуры и вывода ее на индикатор.

```

61 0031 CA 5B 00      JZ   KCLO1      ДА, ЭТО ПЕРВЫЙ РАЗ
62 0034      *
63 0034      ***      НЕ ПЕРВЫЙ РАЗ      *****
64 0034      *
65 0034      ***** (ШАГ 7)
66 0034      *
67 0034 3A 02 10      LDA   KWGHT      КАКАЯ ЦИФРА БЫЛА НАБРАНА???
68 0037 4F      MOV   C, A      РЕГИСТР C=KWGHT
69 0038 3A 03 10      LDA   KCOMP      РЕГИСТР A=KCOMP
70 003B B9      CMP   C      KCOMP=KWGHT???
71 003C CA 46 00      JZ   KCLO2      ДА, ОНИ РАВНЫ!!!
72 003F      *
73 003F      ***** (ШАГ 8)
74 003F      *
75 003F AF      XRA   A      ОНИ НЕ РАВНЫ
76 0040 32 00 10      STA   KTIME      ПЕРЕЗАПИСАТЬ KTIME
77 0043 C3 66 00      JMP   KCLO3      ИДТИ НА ПЕРЕЗАГРУЗКУ АКТИВ. НАБОРА
78 0046      *
79 0046      ***** (ШАГ 9)
80 0046      *
81 0046 3A 00 10      LDA   KTIME      ВЫЗВАТЬ KTIME ИЗ ПАМЯТИ
82 0049 3C      INR   A      KTIME=KTIME+1
83 004A 32 00 10      STA   KTIME      KTIME=KTIME+1
84 004D      *
85 004D      ***** (ШАГ 10)
86 004D      *
87 004D FE 32      CPI   50      KTIME=50???
88 004F C2 66 00      JNZ   KCLO3      ЕЩЕ НЕ 50
89 0052 CD 6F 00      CALL  KORN      ПРОВЕРИТЬ ГОТОВНОСТЬ ПУЛЬТА
90 0055 CD A7 00      CALL  KOUT      ВЫДАТЬ НА ПУЛЬТ
91 0058 C3 1B 00      JMP   SROW      НАЗАД К НАЧАЛУ
92 005B      *
93 005B      ***** (ШАГ 3)
94 005B      *
95 005B 3A 02 10      KCLO1 LDA   KWGHT
96 005E 32 03 10      STA   KCOMP      KCOMP=KWGHT
97 0061      *
98 0061      ***** (ШАГ 4)
99 0061      *
100 0061 3E 01      MVI   A, 01
101 0063 32 00 10      STA   KTIME      УСТАНОВИТЬ KTIME=1
102 0066      *
103 0066      ***** (ШАГ 5)
104 0066      *
105 0066 3E 01      KCLO3 MVI   A, 01
106 0068 32 04 10      STA   KROW      УСТАНОВИТЬ АКТИВ. НАБОР=00000001
107 006B      *
108 006B      ***** (ШАГ 6)
109 006B      *
110 006B C3 1B 00      JMP   SROW      ИДТИ К НАЧАЛУ ПРОГРАММЫ
111 006E      *
112 006E      *
113 006E      *****
114 006E      *
115 006E      *****      НАЧАЛО ПРОГРАММЫ      *****
116 006E      *
117 006E      *****
118 006E      *
119 006E      *
120 006E      *      РЭС .4.22

```

```

121 006E      * ПРОГРАММА ПОДГОТОВКИ ПУЛЬТА
122 006E      *
123 006E      * *****
124 006E      *
125 006E      * ***** (ШАГ 1)
126 006E      *
127 006E 3E 01 KOPN MVI A,01
128 0070 32 04 10 STA KROW KROW=00000001
129 0073 3D DCR A
130 0074 32 05 10 STA NROW NROW=00000000
131 0077      *
132 0077      * ***** (ШАГ 2)
133 0077      *
134 0077 32 00 10 STA KTIME KTIME=0
135 007A      *
136 007A      * ***** (ШАГ 3)
137 007A      *
138 007A CD EB 00 KOPN1 CALL OROW ВЫВЕСТИ НАБОР
139 007D      *
140 007D      * ***** (ШАГ 4)
141 007D      *
142 007D CD DB 00 CALL COLM ВВЕСТИ ДАННЫЕ СТОЛБЦА
143 0080      *
144 0080      * ***** (ШАГ 5)
145 0080      *
146 0080 3A 01 10 LDA CFLAG ВВЕСТИ CFLAG
147 0083 FE 00 CFI 00 CFLAG=0??
148 0085 C2 6E 00 JNZ KOPN ПУЛЬТ ЕЩЕ НЕ СВОБОДЕН
149 0088      *
150 0088      * ***** (ШАГ 9)
151 0088      *
152 0088 3A 05 10 LDA NROW ВВЕСТИ АКТИВНЫЙ НАБОР
153 008B 3C INR A
154 008C 32 05 10 STA NROW ЗАПИСАТЬ АКТИВНЫЙ НАБОР
155 008F      *
156 008F      * ***** (ШАГ 10)
157 008F      *
158 008F FE 05 CFI 05
159 0091 C2 7A 00 JNZ KOPN1 СКАНИРОВАНИЕ НЕ ЗАКОНЧЕНО
160 0094      *
161 0094      * ***** (ШАГ 6)
162 0094      *
163 0094 3A 00 10 LDA KTIME
164 0097 3C INR A
165 0098 32 00 10 STA KTIME
166 009B 47 MOV B,A РЕГИСТР B=KTIME
167 009C      *
168 009C      * ***** (ШАГ 6A)
169 009C      *
170 009C AF XRA A
171 009D 32 05 10 STA NROW ЗНАЧЕНИЕ НАБОРА=0
172 00A0      *
173 00A0      * ***** (ШАГ 7)
174 00A0      *
175 00A0 3E 32 MVI A,50
176 00A2 B8 CMP B KTIME=50???
177 00A3 C2 7A 00 JNZ KOPN1 НЕТ, СКАНИРОВАНИЕ ЕЩЕ РАЗ
178 00A6      *
179 00A6      * ***** (ШАГ 8)
180 00A6      *

```



```

181 00A6 C9          RET          ВЫХОД ИЗ ПОДПРОГРАММЫ ПОДГОТОВКИ
182 00A7          *
183 00A7          *
184 00A7          *
185 00A7          *
186 00A7          * ПОДПРОГРАММА KOUT
187 00A7          *
188 00A7          * ЭТА ПОДПРОГРАММА ВЫВЕДЕТ ДАННЫЕ С ПУЛЬТА НА ДИСПЛЕЙ
189 00A7          *
190 00A7          *
191 00A7          *
192 00A7 3A 02 10 KOUT LDA KWGHT  ДАННЫЕ С ПУЛЬТА В АККУМУЛЯТОРЕ
193 00AA D3 F0      OUT  OF0H
194 00AC D3 F1      OUT  OF1H
195 00AE D3 F2      OUT  OF2H
196 00B0 C9          RET
197 00B1          *
198 00B1          *
199 00B1          *
200 00B1          * ПОДПРОГРАММА KEYW
201 00B1          *
202 00B1          * ЭТА ПОДПРОГРАММА УСТАНОВИТ ЗНАЧЕНИЕ НАЖАТОЙ КЛАВИШИ
203 00B1          *
204 00B1          *
205 00B1          *
206 00B1          * БЛОК-СХЕМА ЭТОЙ ПРОГРАММЫ НА РИС.4.18
207 00B1          *
208 00B1          *
209 00B1          *
210 00B1 AF        KEYW XRA A      ОБНУЛИТЬ ФЛАЖКИ И АККУМУЛЯТОР
211 00B2 3A 01 10 LDA CFLAG      ПРИНЯТЬ ФЛАЖОК СТОЛБЦА ИЗ ПАМЯТИ
212 00B5          *
213 00B5          *
214 00B5          *
215 00B5 06 00      MVI B,00      ОБНУЛИТЬ СЧЕТЧИК
216 00B7          *
217 00B7          *
218 00B7          *
219 00B7 1F        KEYW1 RAR      СДВИНУТЬ ВПРАВО НА 1 БИТ, 0 В D7
220 00B8 FE 00      CPI  00
221 00BA          *
222 00BA          *
223 00BA          *
224 00BA CA C1 00   JZ  KEYW2      РЕГИСТР B=0,1,2,3,4 (ШАГ 6)
225 00BD          *
226 00BD          *
227 00BD          *
228 00BD 04        INR  B
229 00BE C3 B7 00   JMP  KEYW1     УВЕЛИЧИТЬ ЗНАЧ.СЧЕТ. И ИДТИ НА ПОВТОР
230 00C1          *
231 00C1          *
232 00C1          *
233 00C1 AF        KEYW2 XRA A      ОБНУЛИТЬ ФЛАЖКИ
234 00C2 3A 04 10 LDA KROW      ВЫЗВАТЬ АКТИВНЫЙ НАБОР
235 00C5          *
236 00C5          *
237 00C5          *
238 00C5 0E 00      MVI C,00      ОБНУЛИТЬ СЧЕТЧИК T2
239 00C7          *
240 00C7          *

```

241	00C7		*		
242	00C7	1F	KEYW3	RAR	
243	00C8	FE 00		CPI	00
244	00CA		*		
245	00CA		*****	(ШАГ 10)	
246	00CA		*		
247	00CA	CA D1 00		JZ	KEYW4
248	00CD		*		
249	00CD		*****	(ШАГ 14)	
250	00CD		*		
251	00CD	0C		INR	C
252	00CE	C3 C7 00		JMP	KEYW3
253	00D1		*		
254	00D1		*****	(ШАГ 11)	
255	00D1		*		
256	00D1	AF	KEYW4	XRA	A
257	00D2	78		MOV	A,B
258	00D3	17		RAL	
259	00D4	17		RAL	
260	00D5	80		ADD	B*
261	00D6	81		ADD	C
262	00D7		*		
263	00D7		*****	(ШАГ 12)	
264	00D7		*		
265	00D7	32 02 10		STA	KWGHT
266	00DA		*		
267	00DA		*****	(ШАГ 13)	
268	00DA		*		
269	00DA			RET	
270	00DB		*		ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
271	00DB		*****		
272	00DB		*		
273	00DB		*		ПОДПРОГРАММА COLM
274	00DB		*		
275	00DB		*		ЭТА ПОДПРОГРАММА ОПРЕДЕЛИТ АКТИВНЫЙ СТОЛБЕЦ,
276	00DB		*		ЕСЛИ ОН СУЩЕСТВУЕТ
277	00DB		*		
278	00DB		*****		
279	00DB		*		
280	00DB		*		ЭТА ПОДПРОГРАММА-РИС.4.15
281	00DB		*		
282	00DB		*****		
283	00DB		*		
284	00DB		*****	(ШАГ 1)	
285	00DB		*		
286	00DB	AF	COLM	XRA	A
287	00DC	32 01 10		STA	CFLAG
288	00DF		*		УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА=0
289	00DF		*****	(ШАГ 2)	
290	00DF		*		
291	00DF	DB FE		IN	OFEN
292	00E1		*		ВХОДНЫЕ ДАННЫЕ ИЗ ПОРТА FE
293	00E1		*****	(ШАГ 3)	
294	00E1		*		
295	00E1	F6 E0		ORI	OFON
296	00E3		*		УСТАНОВИТЬ БИТЫ D5,D6,D7=1
297	00E3		*****	(ШАГ 4)	
298	00E3		*		
299	00E3	FE FF		CPI	OFEN
300	00E5	CB		RZ	
					ШАГ (4A)

```

301 00E6      *
302 00E6      ***** (ШАГ 5)
303 00E6      *
304 00E6 2F      CMA      ИНВЕРТИРОВАТЬ ДАННЫЕ
305 00E7      *
306 00E7      ***** (ШАГ 6)
307 00E7      *
308 00E7 32 01 10      STA CFLAG      УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
309 00EA C9      RET      ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
310 00EB      *
311 00EB      *****
312 00EB      *
313 00EB      * ПОДПРОГРАММА ВЫВОДА АКТИВНОГО НАБОРА НА ПУЛЬТ
314 00EB      *
315 00EB      * ИМЯ ЭТОЙ ПОДПРОГРАММЫ-OROW
316 00EB      *
317 00EB      * ЭТО РИС.4.12
318 00EB      *
319 00EB      *****
320 00EB      *****
321 00EB      *
322 00EB      * РИС.4.12-ПОДПРОГРАММА К РИС.4.10 В МНЕМОНИКЕ 8080
323 00EB      *
324 00EB      *****
325 00EE      *
326 00EB      *
327 00EB 3A 04 10      OROW LDA KROW      ПОВТОРНЫЙ ВЫЗОВ АКТИВНОГО НАБОРА
328 00EE      *
329 00EE      ***** (ШАГ 2)
330 00EE      *
331 00EE FE 10      CPI 10H      СКАНИРОВАНИЕ НАБОРА ЗАКОНЧЕНО???
332 00FO C2 F8 00      JNZ OROW1      ЕСЛИ НЕТ, ТО ШАГ 2В
333 00F3      *
334 00F3      ***** (ШАГ 2А)
335 00F3      *
336 00F3 3E 01      MVI A, 01H      ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАБОР
337 00F5 C3 F9 00      JMP ST3      ИДТИ НА ШАГ 3
338 00F8      *
339 00F8      ***** (ШАГ 2В)
340 00F8      *
341 00F8 07      OROW1 RLC      СДВИНУТЬ ДАННЫЕ ВЛЕВО, 0 В DO
342 00F9      *
343 00F9      ***** (ШАГ 3)
344 00F9      *
345 00F9 32 04 10      ST3 STA KROW      СОХРАНИТЬ АКТИВНЫЙ НАБОР
346 00FC      *
347 00FC      ***** (ШАГ 4)
348 00FC      *
349 00FC 2F      CMA      ИНВЕРТИРОВАТЬ СЛОВО НАБОРА
350 00FD      *
351 00FD      ***** (ШАГ 5)
352 00FD      *
353 00FD D3 FE      OUT OFEH      ВЫВЕСТИ АКТИВНЫЙ НАБОР В ПОРТ FE
354 00FF      *
355 00FF      ***** (ШАГ 6)
356 00FF      *
357 00FF C9      RET
358 0100      *
359 0100      *
360 0100      *****
361 0100      *
362 0100      END      ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА
      0 ERRORS      22 SYMBOLS

```

```

LF426 T=0003 IS ON CRO0020 USING 00071. BLKS R=1410
0001 1 PAGE 1 F425 WED OCT 03,1979 11:22:42.58
0002
0003
0004
0005 00001 *****
0006 00002 *
0007 00003 * ПРОГРАММА ВВОДА С КЛАВИАТУРЫ И ЗАПИСИ НА ДИСПЛЕЙ
0008 00004 * ДЛЯ 6800
0009 00005 ***** ПРОГРАММИСТ ДЖИМ КОФФРОН *****
0010 00006 *
0011 00007 *
0012 00008
0013 00009 FCOO NAM F425
ORG SFCOO УСТАНОВИТЬ НАЧАЛЬНЫЙ АДРЕС
В ПАМЯТИ

0014 00010 *
0015 00011 * ВО-ПЕРВЫХ ИНИЦИАЛИЗИРУЕМ ВСЕ ПЕРЕМЕННЫЕ
0016 00012 *
0017 00013 FCOO 8E 13FF BEGIN LDS #S13FF ИНИЦИАЛИЗАЦИЯ УКАЗАТЕЛЯ СТЕКА
0018 00014 FCO3 86 00 LDA A #SOO НАЧ. ЗНАЧЕНИЯ ФЛАЖКОВ,
ОВНУЛЕНИЕ АККУМУЛЯТОРА,
ВЫВЕСТИ НОЛЬ НА ДИСПЛЕЙ

0019 00015 FCO5 01 F0 STA A #SFO
0020 00016 FCO7 01 F1 STA A #SF1
0021 00017 FCO9 01 F2 STA A #SF2
0022 00018 FCOB B7 1000 STA A KTIME KTIME=0
0023 00019 FCOE B7 1001 STA A CFLAG CFLAG=0
0024 00020 FC11 B7 1002 STA A KWGHT KWGHT=0
0025 00021 FC14 B7 1003 STA A KCOMP KCOMP=0
0026 00022 FC17 4C INC A АККУМУЛЯТОР=1
0027 00023 FC18 B7 1004 STA A KROW KROW=00000001
0028 00024 *
0029 00025 *
0030 00026 * ТЕПЕРЬ УСТАНОВЛИВАЕМ АДРЕСА ПЕРЕМЕННЫХ
0031 00027 *
0032 00028 1000 KTIME EQU S1000
0033 00029 1001 CFLAG EQU KTIME+1
0034 00030 1002 KWGHT EQU CFLAG+1
0035 00031 1003 KCOMP EQU KWGHT+1
0036 00032 1004 KROW EQU KCOMP+1
0037 00033 1005 NROW EQU KROW+1
0038 00034 *
0039 00035 *
0040 00036 ***** ТЕПЕРЬ-СТАРТ ПРОГРАММЫ *****
0041 00037 *
0042 00038 * НАЧАЛО РИС:4.20
0043 00039 *
0044 00040 FC1B BD FCF2 SROW JSR OROW ВЫВЕСТИ АКТИВНЫЙ НАБОР
0045 00041 FC1E BD FCF5 JSR COLM ВЫЗВАТЬ ПОДПРОГРАММУ COLM
0046 00042 FC21 B6 1001 LDA A CFLAG НЕОБХОДИМО ПРОВЕРИТЬ ФЛАЖОК
СТОЛБЦА
0047 00043 FC24 81 00 CMP A #SOO ПРОВЕРИТЬ АКТИВНОСТЬ СТОЛБЦА
0048 00044 FC26 27 F3 BEQ SROW НЕ АКТИВНЫЙ СТОЛБЕЦ,ПОВТОРИТЬ
ЦИКЛ
0049 00045 FC28 BD FCAD JSR KEYW АКТИВНЫЙ СТОЛБЕЦ,КАКОЙ ИМЕННО?
0050 00046 *
0051 00047 * КОНЕЦ РИС.4.20
0052 00048 *
0053 00049 * НАЧАЛО РИС:4.21
0054 00050 *
0055 00051 ***** ПУЛЬТ АКТИВЕН *****
0056 00052 *
0057 00053 ***** (ШАГ 1)
0058 00054 *

```

Рис. 4.25. Программа на языке ассемблера микропроцессора 6800, предназначенная для ввода информации с клавиатуры и вывода ее на индикатор.

0059 1 PAGE 2 F425 WED OCT 03,1979 11:22:42.58

```
0060
0061
0062
0063 00055 FC2B B6 1000 LDA A KTIME ПЕРЕЗАГРУЗИТЬ KTIME ИЗ ПАМЯТИ
0064 00056 *
0065 00057 *
0066 00058 ***** (ШАГ 2)
0067 00059 *
0068 00060 FC2E 81 00 CMP A #S00 KTIME=0??
0069 00061 FC30 27 25 BEQ KCLO1 ДА, ЭТО ПЕРВЫЙ РАЗ
0070 00062 *
0071 00063 *** НЕ ПЕРВЫЙ РАЗ
0072 00064 *
0073 00065 ***** (ШАГ 7)
0074 00066 *
0075 00067 FC32 F6 1002 LDA B KWGHT КАКАЯ ЦИФРА БЫЛА НАБРАНА???
0076 00068 FC35 B6 1003 LDA A KCOMP РЕГИСТР А=KCOMP
0077 00069 FC38 11 CBA СРАВНИТЬ ДВЕ ВЕЛИЧИНЫ
0078 00070 FC39 27 08 BEQ KCLO2 ДА, ОНИ РАВНЫ
0079 00071 *
0080 00072 ***** (ШАГ 8)
0081 00073 *
0082 00074 FC3B 86 00 LDA A #S00 ОНИ НЕ РАВНЫ
0083 00075 FC3D B7 1000 STA A KTIME ПЕРЕЗАПИСАТЬ KTIME
0084 00076 FC40 7E FC62 JMP KCLO3 ИДТИ НА ПЕРЕЗАГРУЗКУ АКТ. НАБОРА
0085 00077 *
0086 00078 ***** (ШАГ 9)
0087 00079 *
0088 00080 FC43 B6 1000 KCLO2 LDA A KTIME ВЫЗВАТЬ KTIME ИЗ ПАМЯТИ
0089 00081 FC46 4C INC A KTIME=KTIME+1
0090 00082 FC47 B7 1000 STA A *KTIME KTIME=KTIME+1
0091 00083 *
0092 00084 ***** (ШАГ 10)
0093 00085 *
0094 00086 FC4A 81 32 CMP A #50 KTIME=50??
0095 00087 FC4C 26 14 BNE KCLO3 ЕЩЕ НЕ 50
0096 00088 FC4E BD FC6A JSR KOPN ПРОВЕРИТЬ ГОТОВНОСТЬ ПУЛЬТА
0097 00089 FC51 BD FCA3 JSR KOUT ВЫДАТЬ НА ПУЛЬТ
0098 00090 FC54 7E FC1B JMP SROW НАЗАД К НАЧАЛУ
0099 00091 *
0100 00092 ***** (ШАГ 3)
0101 00093 *
0102 00094 FC57 B6 1002 KCLO1 LDA A KWGHT
0103 00095 FC5A B7 1003 STA A KCOMP KCOMP=KWGHT
0104 00096 *
0105 00097 ***** (ШАГ 4)
0106 00098 *
0107 00099 FC5D 86 01 LDA A #S01
0108 00100 FC5F B7 1000 STA A KTIME УСТАНОВИТЬ KTIME=1
0109 00101 *
0110 00102 ***** (ШАГ 5)
0111 00103 *
0112 00104 FC62 86 01 KCLO3 LDA A #S01
0113 00105 FC64 B7 1004 STA A KROW УСТАНОВИТЬ АКТ. НАБ.=00000001
0114 00106 *
0115 00107 ***** (ШАГ 6)
0116 00108 *
0117 00109 FC67 7E FC1B JMP SROW ИДТИ НА НАЧАЛО ПРОГРАММЫ
0118 1 PAGE 3 F425 WED OCT 03,1979 11:22:42.58
```

```

0119
0120
0121
0122 00110
0123 00111
0124 00112
0125 00113
0126 00114
0127 00115
0128 00116
0129 00117
0130 00118
0131 00119
0132 00120
0133 00121
0134 00122
0135 00123
0136 00124
0137 00125
0138 00126 FC6A 86 01 KOPN LDA A #S01
0139 00127 FC6C B7 1004 STA A KROW KROW=00000001
0140 00128 FC6F 4A DEC A
0141 00129 FC70 B7 1005 STA A NROW NROW=00000000
0142 00130
0143 00131
0144 00132
0145 00133 FC73 B7 1000 STA A KTIME KTIME=0
0146 00134
0147 00135
0148 00136
0149 00137 FC76 BD FC72 KOPN1 JSR OROW ВВЕСТИ НАБОР
0150 00138
0151 00139
0152 00140
0153 00141 FC79 BD FCDF JSR COLM ВВЕСТИ ДАННЫЕ СТОЛБЦА
0154 00142
0155 00143
0156 00144
0157 00145 FC7C B6 1001 LDA A CFLAG ВВЕСТИ CFLAG
0158 00146 FC7F 81 00 CMP A #00 CFLAG=0??
0159 00147 FC81 26 E7 BNE KOPN ПУЛЕТ ЕЩЕ НЕ СВОБОДЕН
0160 00148
0161 00149
0162 00150
0163 00151 FC83 B6 1005 LDA A NROW ВВЕСТИ АКТИВНЫЙ НАБОР
0164 00152 FC86 4C INC A
0165 00153 FC87 B7 1005 STA A NROW ЗАПИСАТЬ АКТИВНЫЙ НАБОР
0166 00154
0167 00155
0168 00156
0169 00157 FC8A 81 05 CMP A #05
0170 00158 FC8C 26 E8 BNE KOPN1 СКАНИРОВАНИЕ НЕ ЗАКОНЧЕНО
0171 00159
0172 00160
0173 00161
0174 00162 FC8E B6 1000 LDA A KTIME
0175 00163 FC91 4C INC A
0176 00164 FC92 B7 1000 STA A KTIME
0177 1 PAGE 4 F425 WED OCT 03,1979 11:22:42.58
0178

```

```

0179
0180
0181 00165 FC95 F6 1000 LDA B KTIME РЕГИСТР B=KTIME
0182 00166 *
0183 00167 ***** (ШАГ 6А)
0184 00168 *
0185 00169 FC98 86 00 LDA A #00
0186 00170 FC9A B7 1005 STA A NROW ЗНАЧЕНИЕ НАБОРА=0
0187 00171 *
0188 00172 ***** (ШАГ 7)
0189 00173 *
0190 00174 FC9D 86 32 LDA A #50
0191 00175 FC9F 11 CBA KTIME=50??
0192 00176 FCA0 26 D4 BNE KOPN1 НЕТ, СКАНИРОВАНИЕ ЕЩЕ РАЗ
0193 00177 *
0194 00178 ***** (ШАГ 8)
0195 00179 *
0196 00180 FCA2 39 . RTS ВЫХОД ИЗ ПРОГРАММЫ ПОДГОТОВКИ
0197 00181 *
0198 00182 *
0199 00183 *****
0200 00184 *
0201 00185 * ПОДПРОГРАММА KOUT
0202 00186 *
0203 00187 * ЭТА ПОДПРОГРАММА ВЫВОДИТ ДАННЫЕ С ПУЛЬТА НА ДИСПЛЕЙ
0204 00188 *
0205 00189 *****
0206 00190 *
0207 00191 FCA3 B6 1002 KOUT LDA A KWGHT ДАННЫЕ С ПУЛЬТА В АККУМУЛЯТОРЕ
0208 00192 FCA6 01 F0 STA A #SFO
0209 00193 FCA8 01 F1 STA A #SF1
0210 00194 FCAA 01 F2 STA A #SF2
0211 00195 FCAE 39 RTS
0212 00196 *
0213 00197 *****
0214 00198 *
0215 00199 * ПОДПРОГРАММА KEYW
0216 00200 *
0217 00201 * ЭТА ПОДПРОГРАММА УСТАНОВИТ ЗНАЧЕНИЕ КЛАВИШИ
0218 00202 *
0219 00203 *****
0220 00204 *
0221 00205 ***** БЛОК-СХЕМА ЭТОЙ ПОДПРОГРАММЫ НА РИС.4.18
0222 00206 *
0223 00207 ***** (ШАГ 1)
0224 00208 *
0225 00209 FCAD 86 00 KEYW LDA A #00 НУЛЕВЫЕ ФЛАЖКИ И АККУМУЛЯТОР
0226 00210 FCAF B6 1001 LDA A CFLAG ПРИНЯТЬ ФЛАЖОК СТ. ИЗ ПАМЯТИ
0227 00211 *
0228 00212 ***** (ШАГ 2)
0229 00213 *
0230 00214 FCB2 C6 00 LDA B #00 ОБНУЛИТЬ СЧЕТЧИК
0231 00215 *
0232 00216 ***** (ШАГ 3)
0233 00217 *
0234 00218 FCB4 47 KEYW1 ASR A СДВИНУТЬ ВПРАВО НА 1 БИТ,
О В D7
0235 00219 FCB5 81 00 CMP A #00
0236 1 RAGE 5 F425 WED OCT 03, 1979 11:22:42.58
0237
0238

```

0239					
0240	00220	FCB7 37	PSH B		
0241	00221	*			
0242	00222	***** (ШАГ 4)			
0243	00223	*			
0244	00224	FCB8 27 05	BEQ KEYW2	BREG=0,1,2,3,4 (ШАГ 6)	
0245	00225	*			
0246	00226	FCBA 33	PUL B		
0247	00227	***** (ШАГ 5)			
0248	00228	*			
0249	00229	FCBB 5C	INC B		
0250	00230	FCBC 7E FCB4	JMP KEYW1	УВЕЛИЧИТЬ ЗНАЧ.СЧ. И ИДТИ НА ПОВТОР	
0251	00231	*			
0252	00232	***** (ШАГ 7)			
0253	0233	*			
0254	00234	FCBF 86 00	KEYW2 LDA A #00	ОБНУЛИТЬ ФЛАЖКИ	
0255	00235	FCC1 B6 1004	LDA A KROW	ВЫЗВАТЬ АКТИВНЫЙ НАБОР	
0256	00236	*			
0257	00237	***** (ШАГ 8)			
0258	00238	*			
0259	00239	FCC4 C6 00	LDA B #00	ОБНУЛИТЬ СЧЕТЧИК T2	
0260	00240	*			
0261	00241	***** (ШАГ 9)			
0262	00242	*			
0263	00243	FCC6 47	KEYW3 ASR A	СДВИГ ВПРАВО НА 1 БИТ, 0 В D7	
0264	00244	FCC7 81 00	CMP A #00		
0265	00245	FCC9 37	PSH B		
0266	00246	*			
0267	00247	***** (ШАГ 10)			
0268	00248	*			
0269	00249	FCCA 27 05	BEQ KEYW4	ЕСЛИ 0, ТО ВЫПОЛНЕНО	
0270	00250	*			
0271	00251	FCCC 33	PUL B		
0272	00252	***** (ШАГ 14)			
0273	00253	*			
0274	00254	FCCD 5C	INC B	УВЕЛИЧИТЬ ЗНАЧЕНИЕ СЧЕТЧИКА	
0275	00255	FCCE 7E FCC6	JMP KEYW3	ИДТИ НА ПОВТОР	
0276	00256	*			
0277	00257	***** (ШАГ 11)			
0278	00258	*			
0279	00259	FCD1 33	KEYW4 PUL B	РЕГИСТР B=T2	
0280	00260	FCD2 32	PUL A	АККУМУЛЯТОР=СЧЕТЧИК СТОЛБЦОВ/T1/	
0281	00261	FCD3 37	PSH B		
0282	00262	FCD4 36	PSH A		
0283	00263	FCD5 48	ASL A	T1*2	
0284	00264	FCD6 48	ASL A	T1*4	
0285	00265	FCD7 33	PUL B		
0286	00266	FCD8 1B	ABA	T1*5	
0287	00267	FCD9 33	PUL B		
0288	00268	FCDA 1B	ABA	(T1*5)+T2	
0289	00269	*			
0290	00270	***** (ШАГ 12)			
0291	00271	*			
0292	00272	FCDB B7 1002	STA A KWCHT		
0293	00273	*			
0294	00274	***** (ШАГ 13)			
0295	1	PAGE 6 F425 WEB OCT 03, 1979 11:22:42.58			
0296					
0297					
0298					


```

0299 00275
0300 00276 FCDE 39
0301 00277
0302 00278
0303 00279
0304 00280
0305 00281
0306 00282
0307 00283
0308 00284
0309 00285
0310 00286
0311 00287
0312 00288
0313 00289
0314 00290
0315 00291
0316 00292
0317 00293 FCDF 86 00
0318 00294 FCE1 B7 1001
0319 00295
0320 00296
0321 00297
0322 00298 FCE4 86 FE
0323 00299
0324 00300
0325 00301
0326 00302 FCE6 8A E0
0327 00303
0328 00304
0329 00305
0330 00306 FCE8 81 FF
0331 00307 FCEA 26 01
0332 00308 FCEC 39
0333 00309
0334 00310
0335 00311
0336 00312 FCED 43
0337 00313
0338 00314
0339 00315
0340 00316 FCEE B7 1001
0341 00317 FCF1 39
0342 00318
0343 00319
0344 00320
0345 00321
0346 00322
0347 00323
0348 00324
0349 00325
0350 00326
0351 00327
0352 00328
0353 00329
0354 1 PAGE 7 F425 WED OCT 03,1979 11:22:42.58
0355
0356
0357
0358 00330

```

*
 RTS
 ВЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
 *

 *
 ПОДПРОГРАММА COLM
 *
 * ЭТА ПОДПРОГРАММА ОПРЕДЕЛИТ АКТИВНЫЙ СТОЛБЕЦ,
 * ЕСЛИ ОН ЕСТЬ.
 *

 *
 *
 *

 *
 ***** (ШАГ 1)
 *
 COLM LDA A #00
 STA A CFLAG УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА=0
 *
 ***** (ШАГ 2)
 *
 LDA A #SFE ВВЕСТИ ДАННЫЕ ИЗ ПОРТА FE
 *
 ***** (ШАГ 3)
 *
 ORA A #SEO УСТАНОВИТЬ БИТЫ D5,D6,D7=1
 *
 ***** (ШАГ 4)
 *
 CMP A #SFF
 BNE NEXT
 RTS ШАГ (4A)
 *
 ***** (ШАГ 5)
 *
 NEXT COM A ИНВЕРТИРОВАТЬ ДАННЫЕ
 *
 ***** (ШАГ 6)
 *
 STA A CFLAG УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
 RTS ВЙТИ ИЗ ПОДПРОГРАММЫ
 *

 *
 * ПОДПРОГРАММА ВЫВОДА АКТИВНОГО НАБОРА НА ПУЛЬТ
 *
 * ИМЯ ЭТОЙ ПОДПРОГРАММЫ OROW
 *
 *
 *

 *
 *

```

00359 00331 *
00360 00332 *****
00361 00333 *
00362 00334 *
00363 00335 FCF2 B6 1004 OROW LDA A KROW ВЫЗВАТЬ АКТИВНЫЙ НАБОР
00364 00336 *
00365 00337 * ШАГ 2
00366 00338 *
00367 00339 FCF5 81 10 CMP A #S10 СКАНИРОВАНИЕ НАБОРА ЗАКОНЧЕНО
00368 00340 FCF7 26 05 BNE OROW1 ЕСЛИ НЕТ, ТО ШАГ 2В
00369 00341 *
00370 00342 * ШАГ 2А
00371 00343 *
00372 00344 FCF9 86 01 LDA A #S01 ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАБОР
00373 00345 FCFB 7E FCF7 JMP ST3 ИДТИ НА ШАГ 3
00374 00346 *
00375 00347 * ШАГ 2В
00376 00348 *
00377 00349 FCFE 48 OROW1 ASL A СДВИНУТЬ ДАННЫЕ ВЛЕВО, О В ДО
00378 00350 *
00379 00351 * ШАГ 3
00380 00352 *
00381 00353 FCFE B7 1004 ST3 STA A KROW СОХРАНИТЬ АКТ.НАБОР В ПАМЯТИ
00382 00354 *
00383 00355 * ШАГ 4
00384 00356 *
00385 00357 FDO2 43 COM A ИНВЕРТИРОВАТЬ СЛОВО НАБОРА
00386 00358 *
00387 00359 * ШАГ 5
00388 00360 *
00389 00361 FDO3 01 FE STA A #SFF ВЫВЕСТИ АКТИВНЫЙ НАБОР В ПОРТФЕ
00390 00362 *
00391 00363 * ШАГ 6
00392 00364 *
00393 00365 FDO5 39 RTS
00394 00366 *
00395 00367 ***** УСТАНОВИМ ВЕКТОР ПОВТОРНОГО СТАРТА *****
00396 00368 *
00397 00369 FFFE ORG $FFFF
00398 00370 FFFE FCB0 FDB BEGIN
00399 00371 *
00400 00372 *
00401 00373 *
00402 00374 *****
00403 00375 *
00404 00376 END ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА
00405 1 PAGE 8 F425 WED OCT 03, 1979 11:22:42.58
00406
00407
00408
0409 BEGIN FCB0 00013 00370
0410 CFLAG 1001 00029 00316 00294 00210 00145 00042 00030 00019
0411 COLM FCBF 00293 00141 00041
0412 KCL01 FC57 00094 00061
0413 KCL02 FC43 00080 00070
0414 KCL03 FC62 00104 00087 00076
0415 KCOMP 1003 00031 00095 00068 00032 00021
0416 KEYW FCBF 00209 00045
0417 KEYW1 FCB4 00218 00230
0418 KEYW2 FCBF 00234 00224

```

```

0419 KEYW3 FCG6 00243 00255
0420 KEYW4 FCD1 00259 00249
0421 KOPN FC6A 00126 00147 00088
0422 KOPN1 FC76 00137 00176 00158
0423 KOUT FCA3 00191 00089
0424 KROW 1004 00032 00353 00335 00235 00127 00105 00033 00023
0425 KTIME 1000 00028 00165 00164 00162 00133 00100 00082 00080 00075 00055
0426
0427 KWGHT 1002 00030 00272 00191 00094 00067 00031 00020
0428 NEXT FCED 00312 00307
0429 NROW 1005 00033 00170 00153 00151 00129
0430 OROW FCF2 00335 00137 00040
0431 OROW1 FCFE 00349 00340
0432 SROW FC1B 00040 00109 00090 00044
0433 ST3 FCFE 00353 00345

```

4.13. Выводы

В этой главе рассмотрены принципы ввода данных с клавишного пульта и вывода информации на цифровой индикатор. Рассмотренная нами микропроцессорная система не ориентировалась на выполнение какой-либо практической задачи. Однако если принципы работы системы понятны, то к средствам проектирования добавилось еще одно, которое можно использовать при проектировании небольших микропроцессорных систем. В гл. 8 будет показано, как применять рассмотренные здесь принципы проектирования для решения практической задачи — автоматического программирования ППЗУ. При этом будут использоваться клавишный пульт и индикатор, рассмотренные в данной главе. Конечно, придется добавить как программные, так и аппаратные средства. Однако общие принципы организации функционирования клавишного пульта и индикатора будут непосредственно применяться при проектировании. В гл. 5 нам предстоит рассмотреть методы отладки небольших микропроцессорных систем и методы поиска неисправностей в отказавшем оборудовании. Прежде чем переходить к изучению материала следующей главы, следует хорошо разобраться в вопросах, изложенных в данной главе.

ПРИМЕНЕНИЕ МЕТОДА ТЕСТИРОВАНИЯ СТАТИЧЕСКИМИ СИГНАЛАМИ ДЛЯ ОТЛАДКИ АППАРАТНЫХ СРЕДСТВ МИКРОПРОЦЕССОРНЫХ СИСТЕМ

В настоящей главе рассматривается метод отладки аппаратных средств микропроцессорных систем. Этот метод может с одинаковым успехом применяться и для отладки опытных образцов только что сконструированных систем, и для систем, которые были в эксплуатации, но по каким-либо причинам утратили работоспособность.

Метод, используемый для отладки аппаратных средств, носит название «тестирование статическими сигналами». Этот метод разработан автором. Идея его впервые была изложена в книге *Understanding and Troubleshooting the Microprocessor*, Prentice Hall, 1980. Рассмотрим, как используется метод тестирования статическими сигналами для поиска неисправностей в аппаратных средствах микропроцессорных систем.

5.1. Идея метода тестирования статическими сигналами

В первых четырех главах книги рассматривались различные вопросы построения микропроцессорных систем. В гл. 4 фактически выполнялось проектирование системного программного обеспечения для систем, работающих под управлением микропроцессора. Основываясь на материалах указанных глав, точно укажем состав технических средств микропроцессорных систем.

В микропроцессорные системы входит восемь следующих основных элементов:

1. Микропроцессор.
2. Адресная шина.
3. Двухнаправленная шина данных.
4. Шина управления с логическими схемами и буферами.
5. ПЗУ.
6. ОЗУ.
7. Устройства ввода.
8. Устройства вывода.

Отметим, что элементы 2—8 работают под управлением микропроцессора. Действительно, микропроцессор генерирует управляющие сигналы, которые используются для управления работой ПЗУ, ОЗУ и устройств ввода-вывода. Буферы шин

лишь временно сохраняют двоичную информацию, появляющуюся на выходах микропроцессора. Все это является основанием к применению такого мощного средства отладки аппаратных средств, как метод тестирования статическими сигналами.

Теперь рассмотрим характеристики основных блоков аппаратных средств. Все эти блоки имеют статическую природу, т. е. они не требуют периодического обновления в них информации с помощью специального механизма. Системные ОЗУ, которые предполагается использовать, являются статическими, а не динамическими. Во многих небольших микропроцессорных системах применяются статические ОЗУ.

Приведем здесь два положения, касающиеся аппаратных средств, которые полезно знать:

1. Все периферийные аппаратные средства, такие, как ПЗУ, ОЗУ и устройства ввода-вывода, управляются микропроцессором.

2. Природа всех управляемых блоков аппаратных средств статическая.

Рассмотрим, какую роль играют эти два положения при отладке аппаратных средств с помощью устройства тестирования статическими сигналами. В качестве примера разберем порядок выполнения операции чтения данных из ПЗУ. Чтобы прочитать данные из ПЗУ, микропроцессор должен выполнить следующие четыре действия:

1. Подать адрес на адресную шину.
2. Подготовить шину данных к приему данных.
3. Подать сигнал ЧТЕНИЕ и сигнал выбора кристалла.
4. Снять сигнал ЧТЕНИЕ и сигнал выбора кристалла.

Проведем подробный анализ каждого из перечисленных действий. В ходе анализа постараемся выявить, насколько оправдано использование тестирования статическими сигналами.

Выше указывалось, что, выполняя операцию чтения данных из ПЗУ, микропроцессор должен подать на адресную шину определенный адрес. Как используется этот адрес? Он служит и для выбора модуля ПЗУ, и для указания ячейки в выбранном модуле, в которой содержится нужное 8-разрядное слово данных. Как же выполняются аппаратные средства указанный процесс выборки? Некоторые адресные линии подключены к дешифратору адреса, который обеспечивает формирование сигнала выбора памяти. Другие адресные линии непосредственно связаны с линиями ввода адреса памяти ПЗУ.

Напомним, что выбор кристалла памяти в системе и выбор адреса ячейки в кристалле осуществляются дешифраторами, выполненными в виде комбинационных логических схем. На адресные линии можно подать некоторый адрес и оставить его неизменным в течение продолжительного периода времени. При этом дешифратор адреса будет всегда вырабатывать один и

тот же выходной сигнал. По существу *дешифрирование адреса представляет собой не зависящую от времени функцию*. Сигналы управления, используемые для подготовки шины данных к приему данных, применяются и для подачи сигнала чтения из памяти. И в этом случае сигналы шины управления образуются с помощью дешифрирования определенных выходных сигналов микропроцессора. Здесь сигналы управления образуются аналогично тому, как формируются сигналы выбора памяти, которые получались посредством дешифрирования кодов, поступающих по определенным адресным линиям. Сигналы шины управления формируются путем дешифрирования определенных сигналов управления, вырабатываемых микропроцессором. Существенно, что сигналы управления на выходах микропроцессора можно сохранять в определенном состоянии в течение длительного промежутка времени, а сигналы шины управления при этом будут оставаться неизменно правильными.

Снятие сигнала ЧТЕНИЕ и сигнала выбора памяти — четвертое основное действие, которым завершается операция чтения данных из ПЗУ. При выполнении его, так же как и при выполнении 3-го действия, шина данных используется для передачи сигналов управления, генерируемых микропроцессором.

Из проведенного анализа следует, что микропроцессор действительно является контроллером системы, который способен очень быстро выполнять статические операции. Поскольку каждая статическая операция протекает очень быстро, мы обычно забываем о статической природе систем, управляемых микропроцессором. Благодаря статическому характеру систем для отладки аппаратных средств можно использовать очень эффективный метод — тестирование статическими сигналами.

5.2. Аппаратные средства устройства тестирования статическими сигналами

В общем виде наши рассуждения подчинены схеме, выражаемой следующими словами: «Если мы сделаем это в системе, то произойдет следующее...» Если мы подадим адреса и сигналы управления на определенные схемы системы, то мы сможем выполнить отладку системы. Рассмотрим подробно, какие аппаратные средства требуются для образования сигналов, необходимых для выполнения тестирования. Рассматриваемый здесь тип ввода сигналов подобен способу, который применяется при отладке аналоговых схем.

5.3. Адресные линии

Сначала обсудим, каким образом могут быть использованы адресные линии в системе. Если удалить микропроцессор из системы, то можно подключить кабель к разъему, предназначенно-

Микропроцессор отключается,
и подключается
штепсельный разъем кабеля
устройства тестирования

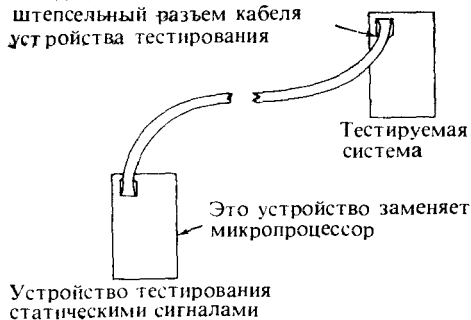


Рис. 5.1. Схема подключения устройства тестирования статическими сигналами к проверяемой микропроцессорной системе.

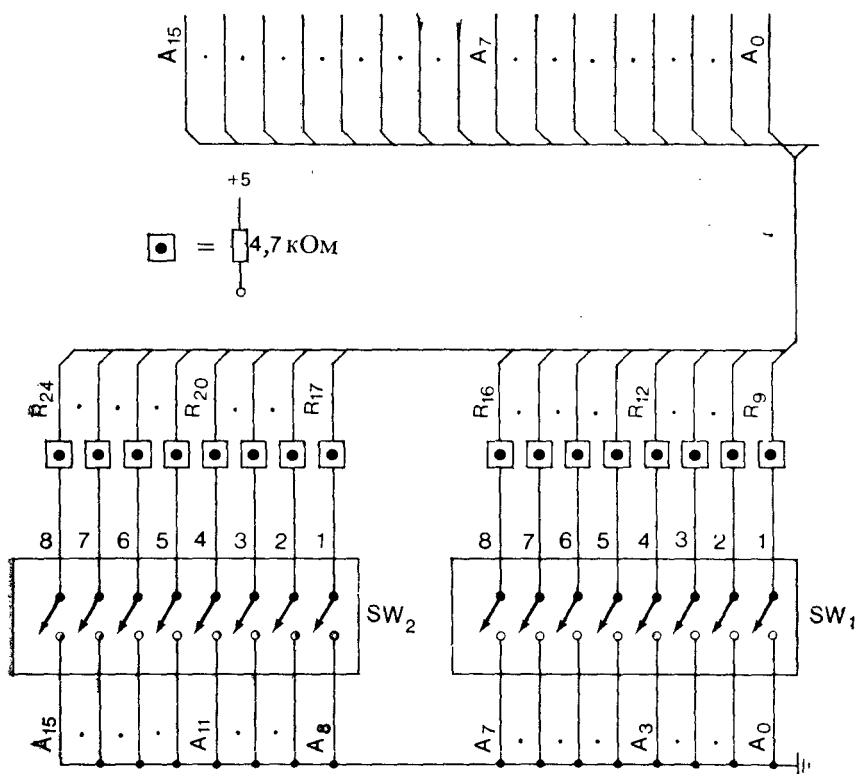


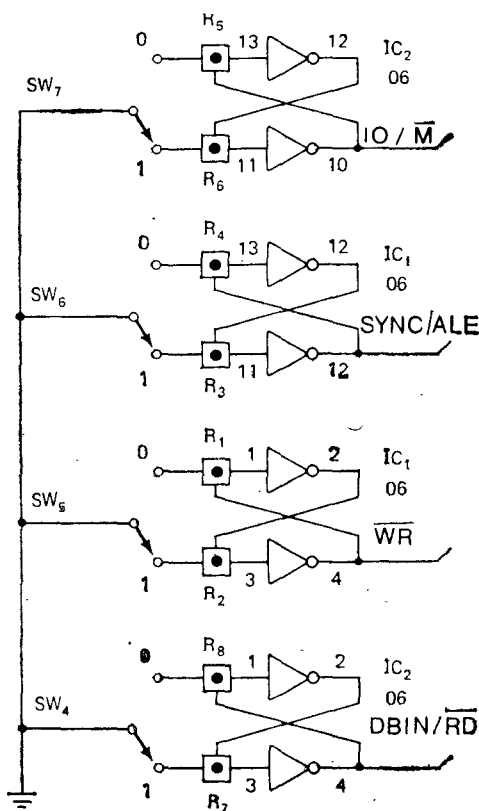
Рис. 5.2. Схема включения адресных тумблеров устройства тестирования статическими сигналами.

му для подключения микропроцессора (рис. 5.1). Тогда к другому концу этого кабеля можно подключить группу тумблеров. Каждый тумблер будет соответствовать определенной линии адреса микропроцессора. На рис. 5.2 представлена схема подключения тумблеров ко всем адресным линиям.

5.4. Сигналы управления

Для имитации сигналов управления микропроцессора можно воспользоваться группой тумблеров, подобных тумблерам адресных линий. Таких тумблеров в группе должно быть ровно

Рис. 5.3. Схема генерации сигналов управления.



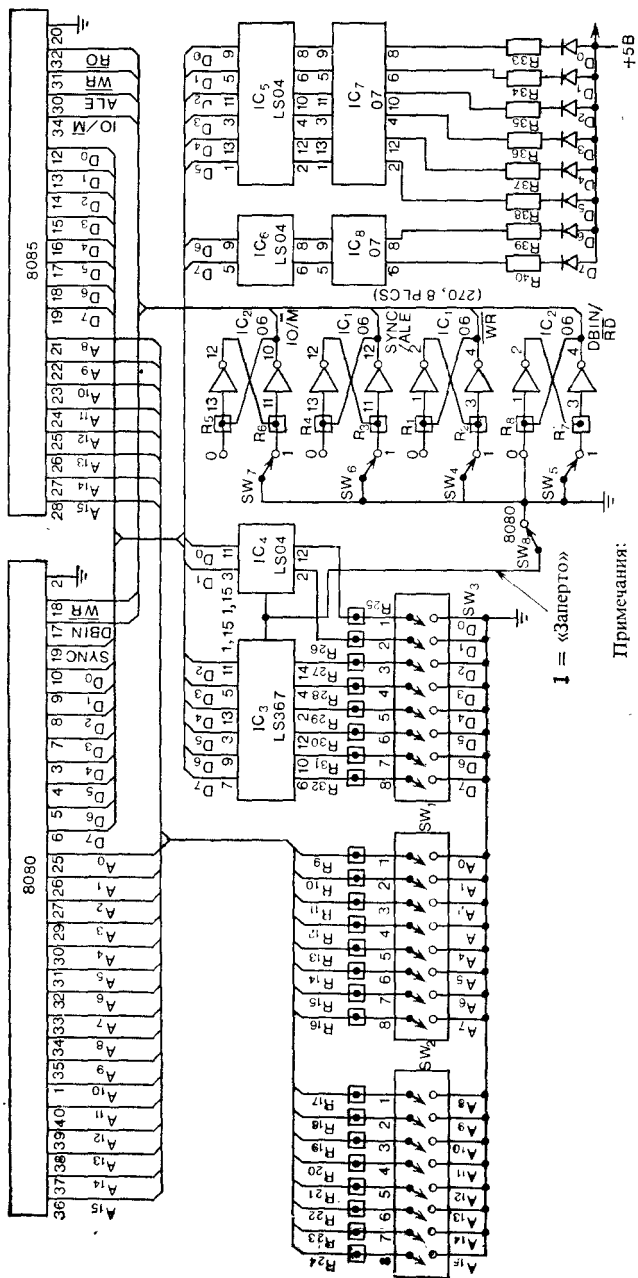
столько, сколько используется выходов микропроцессора для подачи сигналов управления. Все особенности данного вопроса будут раскрыты при проектировании устройства тестирования

статическими сигналами, предназначенного для микропроцессоров 8080 и 8085. Схема генерации сигналов управления представлена на рис. 5.3.

5.5. Линии шины данных

Задача формирования сигналов устройством тестирования статическими сигналами для шины данных отличается от задачи формирования сигналов для адресной шины и шины управления. Отличие обусловлено тем, что шина данных является двунаправленной, а не однонаправленной. Таким образом, необходимо иметь схему для генерации данных, подаваемых на шину данных, и, кроме того, следует предусмотреть дополнительную схему, обеспечивающую управление размещением данных на шине данных, которое осуществляется другим источником. Эти две различные функции точно соответствуют тем, которые присущи микропроцессору при взаимодействии с шиной данных.

В различных микропроцессорах для определения режима использования шины данных — ввод данных или вывод данных — применяются различные линии. Например, микропроцессорная система 8080 работает в режиме чтения данных, когда сигнал DBIN имеет логическое значение 1. В системе 8085 устанавливается режим чтения, когда сигнал $\overline{RD}$ принимает логическое значение 0. Таким образом, для различных типов микропроцессорных систем необходимо по-разному реализовывать аппаратные средства тестирования статическими сигналами. На рис. 5.4 приведены схемы, соответствующие различным режимам использования шины данных. На рис. 5.5 изображена схема аппаратной реализации функций двунаправленной шины данных в устройстве тестирования статическими сигналами. Поясним принцип действия этой схемы. Когда сигнал управления будет иметь логическое значение 0, данные, определяемые положениями тумблеров и индицируемые светоизлучающими диодами, будут поданы на шину данных. В этом режиме устройство тестирования или микропроцессор выводят данные на шину данных микропроцессорной системы. Если на вход сигнала управления подается уровень логической 1, то данные, индицируемые светоизлучающими диодами, поступают на шину данных от какого-либо блока системы. В этом режиме устройство тестирования, имитирующее микропроцессор, вводит данные. Такой режим аналогичен рабочему режиму, в котором микропроцессор запрашивает данные из памяти или от устройств ввода-вывода. На рис. 5.6 представлена схема устройства тестирования статическими сигналами, предназначенного для проверки микропроцессорных систем 8080 и 8085. Она является реальным примером аппаратуры, которая действительно необходима для реализации тестирования статическими сигналами.



Примечания:

1. Все резисторы имеют мощность 0,25 Вт, 5 %
2. □ Резисторы R_1 - R_{32} имеют сопротивление 4,7 кОм
3. Резисторы R_{33} — R_{40} имеют сопротивление 270 Ом

Рис. 5.6. Схема устройства тестирования статических сигналов, предназначенного для микропроцессоров 8080 и 8085.

5.6. Применение устройства тестирования статическими сигналами

На рис. 5.7 показана схема подключения устройства тестирования статическими сигналами к микропроцессору Z80. Устройство тестирования будет использоваться для контроля и отладки аппаратных средств системы, описанной в гл. 4. Схема этой системы изображена на рис. 5.8а—5.8в. Используя устройство тестирования статическими сигналами, пробник, осциллограф или вольтметр, можно систематически и эффективно проверять работоспособность каждой секции аппаратных средств системы. Представляемые здесь методы использовались на практике в промышленности и оказались эффективными.

Рассматриваемые методы отладки являются основополагающими. Они весьма полезны для начинающих, так как экономичны и просты.

5.7. Выбор точки начала контроля

Независимо от типа используемого микропроцессора система всегда предъявляет определенные требования к источнику питания. Поэтому контроль системы целесообразно начать именно с источника питания. Чтобы убедиться, что все уровни напряжений питания находятся в допустимых пределах, следует измерить их с помощью вольтметра. Удостоверившись в том, что питающие напряжения отвечают требованиям, следует приступить к проверке генератора тактовых импульсов. Если в системе использованы системные генераторы тактовых импульсов, то, чтобы определить, удовлетворяют ли их характеристики спецификациям, следует воспользоваться осциллографом. Спецификации определяют верхние и нижние границы для всех временных и амплитудных характеристик генераторов тактовых импульсов. Удостоверившись, что все временные и амплитудные характеристики генератора тактовых импульсов удовлетворяют спецификациям, можно продолжить контроль. При выполнении очередных шагов контроля всегда должна существовать уверенность, что на соответствующие входы микропроцессора подаются номинальные напряжения питания и правильные последовательности синхронизирующих импульсов.

Следующий шаг в микропроцессорах разного типа может выполняться по-разному. Рассмотрим особенности его выполнения для микропроцессора Z80. После детального обсуждения принципов и порядка контроля этого микропроцессора будет не трудно приспособиться к особенностям других микропроцессоров. Рассмотрим, как будет функционировать контролируемая система. Мы знаем, что микропроцессор будет работать как системный контроллер, т. е. он будет непрерывно выполнять про-

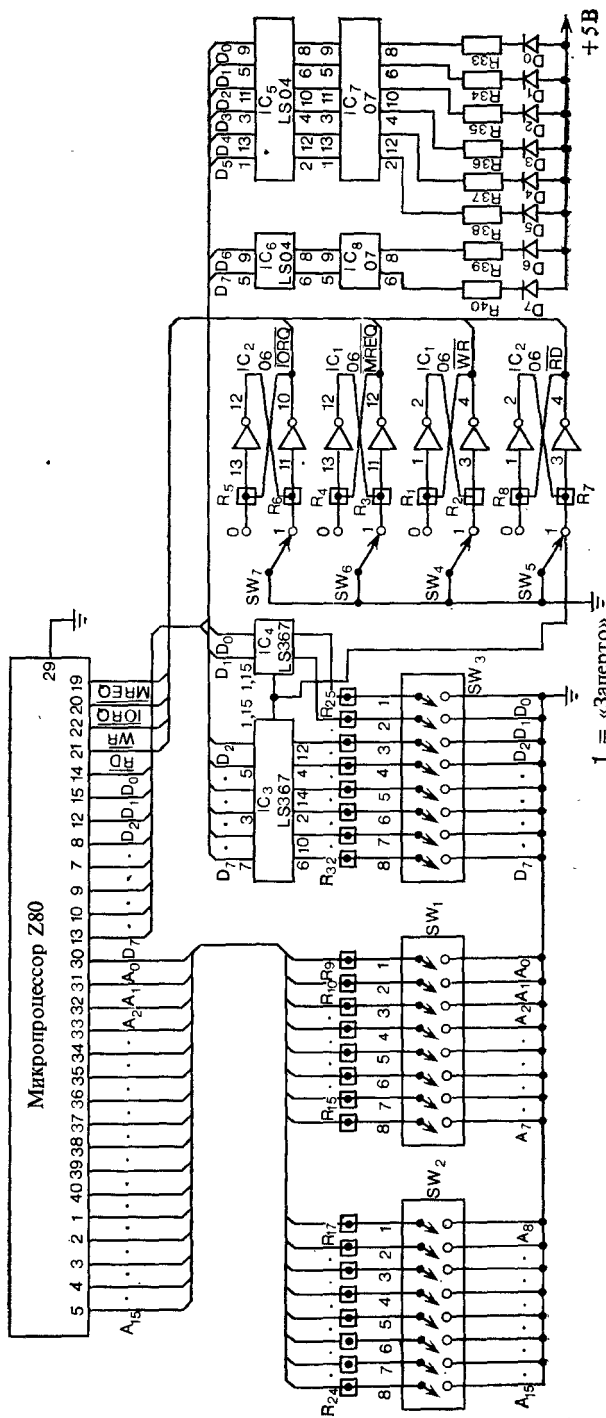


Рис. 5.7. Схема подключения устройства тестирования к микропроцессору Z80.



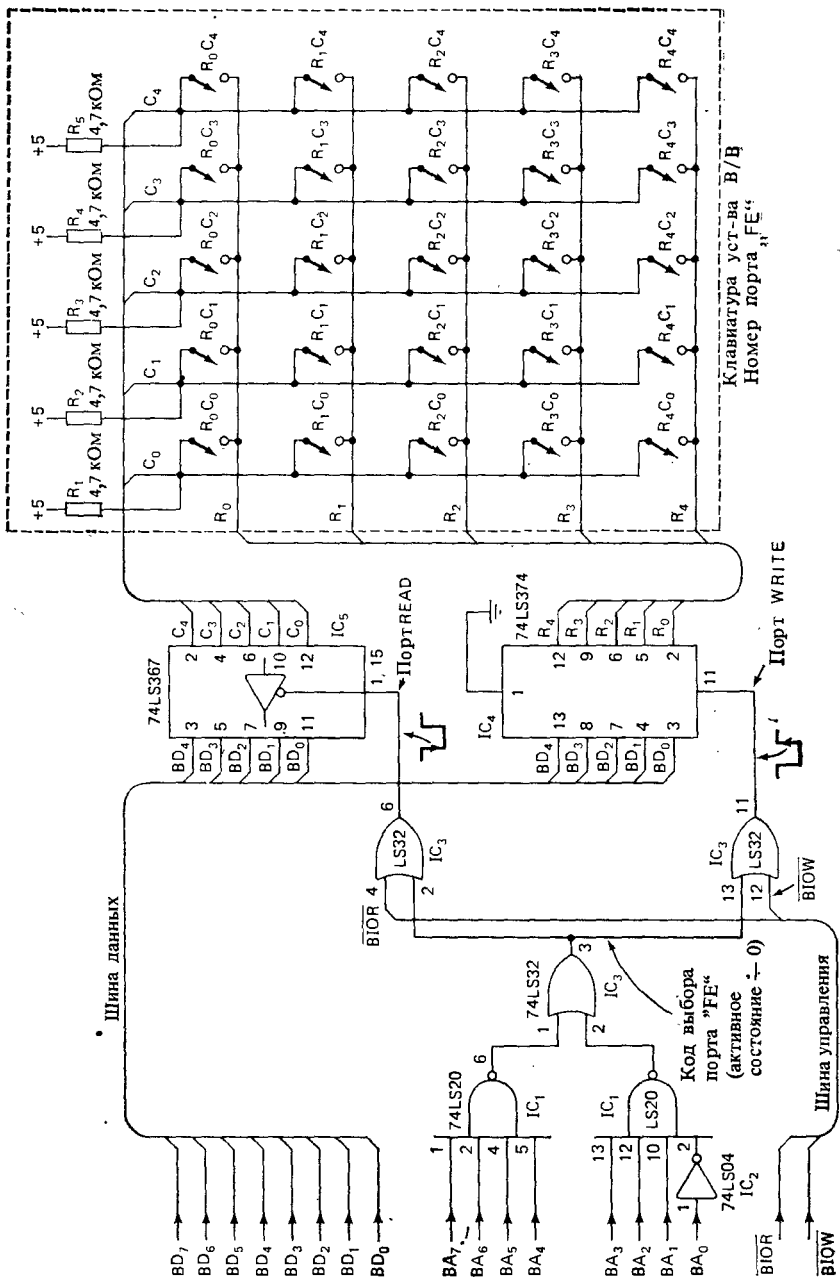


Рис. 5.86. Функциональная схема интерфейса клавишного устройства ввода микропроцессорной системы.

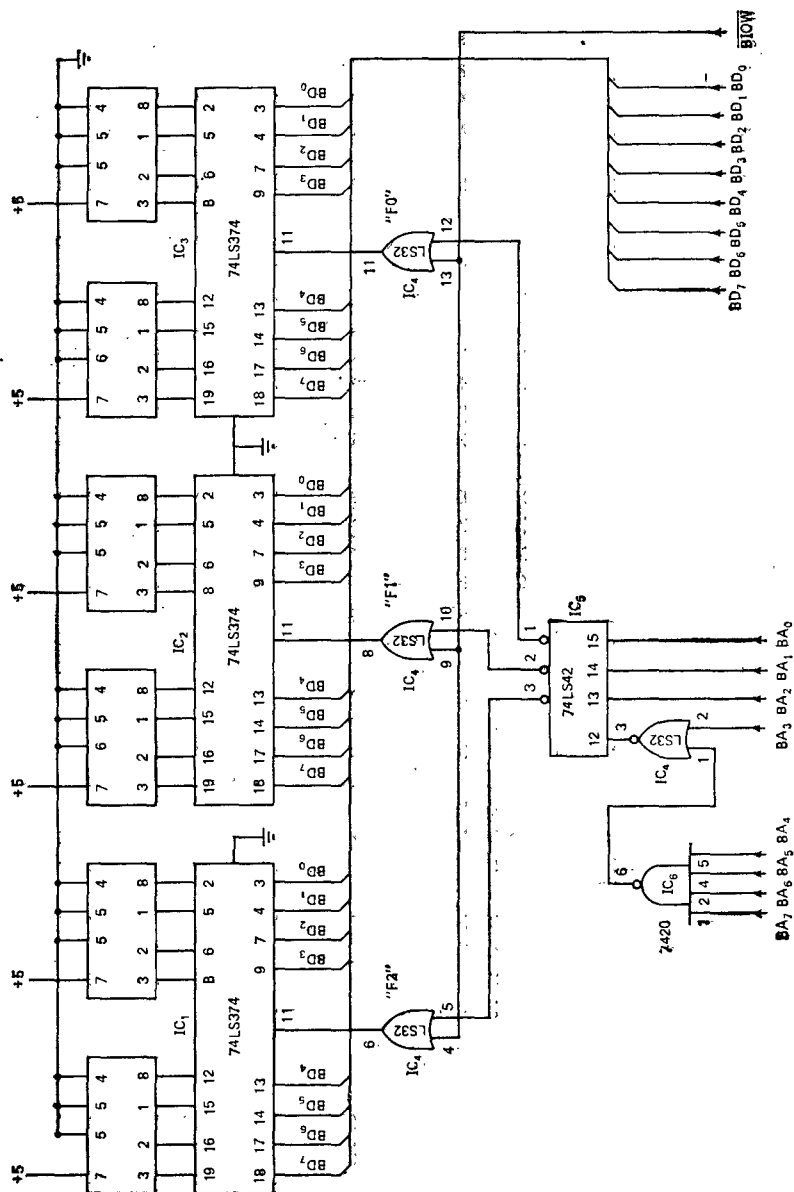


Рис. 5.8в. Функциональная схема интерфейса выходного индикатора микропроцессорной системы.

грамму без прерываний или перехода в состояние ожидания. Кроме того, в состав системы не входят устройства, работающие в режиме прямого доступа к памяти. Входы микропроцессора, соответствующие перечисленным режимам, заблокированы, т. е. они находятся в нерабочем состоянии, однако на этих входах необходимо проверить уровни напряжений. Напомним, что на выходы 24, 16, 17 и 25 микропроцессора Z80 должен подаваться уровень, соответствующий логическому значению 1.

Когда ключ сброса системы, т. е. установки системы в начальное состояние, включен, на входе «сброс» — в микропроцессоре Z80 ему соответствует вывод 26 — должен быть уровень логического 0. Необходимо убедиться, что при переключении ключа сброса системы на входе «сброс» устанавливается как состояние логического 0, так и состояние логической 1. В нормальном режиме работы микропроцессора, когда он работает по программе, на входе «сброс» (вывод 26) должно быть состояние логической 1.

У различных микропроцессоров имеется разное число выходов сигналов управления и по-разному определены логические состояния на этих выходах. Необходимо хорошо знать назначение выводов и логические значения сигналов управления, вырабатываемых каждым микропроцессором.

Итак, мы установили, что в системе созданы все условия для нормальной работы микропроцессора. Теперь микропроцессор можно удалить из системы. Но удалять из системы логические блоки без предварительного выключения питания нельзя. Если питание не выключено, интегральные схемы могут быть повреждены. В этом случае поиск неисправностей или процесс отладки вряд ли завершится успешно.

После удаления микропроцессора на его место в соответствии со схемой, показанной на рис. 5.1, подключается устройство тестирования.

Теперь, когда устройство тестирования статическими сигналами подключено к системе, можно приступить к проверке системы. Ниже рассмотрим методику пошагового тестирования аппаратных средств микропроцессорных систем.

5.8. Проверка адресной шины

Проверяя адресную шину, нужно убедиться, что все адресные линии соединяются с теми точками системы, с которыми должны быть соединены. Далее проверим правильность работы дешифрирующих схем, т. е. убедимся в том, что при дешифрировании не возникает ошибок ни из-за возможных дефектов в аппаратных средствах, ни из-за оплошностей при проектировании системы. Адресная информация поступает в систему с выходов микропроцессора, к которым подключается шина адреса. Сле-

дующая процедура является основной для контроля шины адреса. Размеры этой процедуры и ее трудоемкость велики. Но она будет проверять всю адресную шину. Причем время выполнения этой процедуры уменьшится после приобретения определенных навыков.

Процедура:

1. Установить с помощью адресных тумблеров устройства тестирования статическими сигналами состояние логического 0 на всех адресных линиях.

2. Установить тумблер адресной линии A_0 на устройстве тестирования в состояние логической 1. Тем самым будет установлено состояние логической 1 на выводе № 30 панели микропроцессора Z80.

3. Проверить состояние вывода 13 схемы 74LS367, обозначенной на рис. 5.8а через IC_1 . Эта схема является буфером адреса. На указанном выходе должно быть состояние логической 1.

4. После обнаружения на 13-м выводе схемы 74LS367 состояния логической 1 следует установить адресный тумблер A_0 в состояние логического 0 и снова проверить состояние вывода 13, на котором сейчас должен быть уровень логического 0. Успешное выполнение указанных проверок позволяет убедиться в правильности реакции вывода 13 на изменения логического состояния адресного тумблера A_0 .

5. Если вывод 13 схемы IC_1 правильно реагирует на изменения логического состояния адресного тумблера A_0 , то можно заключить, что схема 74LS367 работает исправно и что соединение микропроцессора с системой выполнено правильно. Теперь можно переходить прямо к шагу 7.

6. Если на выходе схемы IC_1 74LS367 в ответ на изменение положения адресного тумблера A_0 не устанавливается ожидаемое логическое состояние, то нельзя уверенно сказать, дефектна ли схема IC_1 74LS376 или ее выход перегружен. Вспомним, что сигналы по шине адреса параллельно идут к нескольким точкам системы. Любая из этих точек может обусловить перегрузку рассматриваемого нами выхода схемы IC_1 .

6а. Если при тестировании на выходе схемы IC_1 обнаруживаются неправильные логические состояния, то следует проверить правильность изменения логических уровней на входе (вывод 14) схемы IC_1 при изменениях логических состояний на выводе 30 микропроцессора.

6б. Если на входе схемы IC_1 (вывод 14) зафиксированы правильные логические состояния, а на выходе (вывод 13) — ошибочные, т. е. не соответствующие логическим состояниям вывода 30, то для определения причины неисправности можно исполь-

зовать стандартные методы поиска неисправностей в цифровых схемах¹⁾.

7. Допустим, что на выводе 13 схемы IC₁ при тестировании были зафиксированы ожидаемые логические состояния. Далее следует выполнить проверки, аналогичные описанным выше во всех точках системы, соединенных с этим выходом. Таким образом мы убедимся, что выход буфера адреса BA₀ корректно связан со всеми определенными точками системы.

7а. В микропроцессорной системе Z80, снова применяя устройство тестирования статическими сигналами, теперь нужно проверить правильность подачи сигнала на адресные входы BA₀ всех ПЗУ и ОЗУ системы. Переключая тумблер A₀ на устройстве тестирования, следует убедиться, что на адресном входе BA₀ ПЗУ₀ (вывод 8, IC₁₄), ПЗУ₁ (вывод 8, IC₁₅), ОЗУ (вывод 5, IC₁₇, IC₁₈, IC₁₉ и IC₂₀) устанавливаются логические состояния, соответствующие положениям тумблера A₀.

7б. На этом шаге будем проверять адресные входы устройств ввода-вывода. Адресный вход BA₀ клавишного пульта связан с выводом 1 схемы IC₂ (см. рис. 5.8б). Убедимся, что этот адресный вход будет реагировать корректно на изменения выходного сигнала устройства тестирования статическими сигналами. Адресный вход BA₀ индикатора связан с выводом 15 схемы IC₅ (см. рис. 5.8в). И снова убеждаемся в том, что логические состояния в контролируемой точке правильно изменяются при изменениях сигнала на выходе устройства тестирования.

8. Шаги 2—7 следует повторить для всех выходов адресных линий микропроцессора Z80.

Теперь, по-видимому, стало ясно, что при использовании устройства тестирования статическими сигналами для отладки сложных микропроцессорных систем могут быть применены стандартные методы поиска неисправностей в цифровых схемах. Мы не обогатили искусство поиска неисправностей новыми приемами, а лишь рассмотрели новые приложения старых методов.

9. Приступим к проверке линий выбора памяти. Сигналы выбора памяти формируются схемой 74LS42, имеющей на рис. 5.8а обозначение IC₁₁. Входами этой схемы являются адресные линии BA₁₀, BA₁₁, BA₁₂ и BA₁₃. На выходах схемы 74LS42 образуются сигналы выбора памяти ПЗУ и ОЗУ. В системе Z80 используются ПЗУ объемом в 2К, однако допускается расширение ПЗУ до 4К. Кроме того, используется ОЗУ объемом в 2К. Объем ОЗУ можно довести до 3К. Схема распределения памяти показана на рис. 5.9. Используя схему распределения памяти, можно легко определить правильность выбора памяти.

¹⁾ Coffron J. W., *Getting Started in Digital Troubleshooting*, Reston Publishing Co., Reston, VA, 1979.

10. Установить с помощью адресных ключей A_{10} — A_{13} коды, показанные на рис. 5.9, и проверить состояния выходов 1, 2, 3, 4, 5, 6, 7 схемы 74LS42. Эта проверка осуществляется с целью определения правильности реакции выходов на изменения кодов

Диапазон адресов	Функция
0000—03FF	ПЗУ ₀
0400—07FF	ПЗУ ₁
0800—0BFF	ПЗУ ₂ Не использована
0C00—0FFF	ПЗУ ₃ Не использована
1000—13FF	ОЗУ ₁
1400—17FF	ОЗУ ₂
1800—1BFF	ОЗУ ₃ Не использована

Рис. 5.9. Схема распределения памяти тестируемой микропроцессорной системы.

адреса. В правильности логических состояний выходов схемы IC_{11} можно убедиться с помощью таблицы соответствия входных и выходных сигналов, представленной на рис. 5.10.

Выход схемы IC_{11}							BA_{13}	BA_{12}	BA_{11}	BA_{10}
1	2	3	4	5	6	7				
0	1	1	1	1	1	1	0	0	0	0
1	0	1	1	1	1	1	0	0	0	1
1	1	0	1	1	1	1	0	0	0	0
1	1	1	0	1	1	1	0	0	0	1
1	1	1	1	0	1	1	0	1	1	0
1	1	1	1	1	0	1	0	1	1	1
1	1	1	1	1	1	0	0	1	1	0

Рис. 5.10. Логические состояния выходов системы IC_{11} , соответствующие логическим состояниям выходов BA_{10} , BA_{11} , BA_{12} и BA_{13} буфера адреса.

11а. Выходы дешифратора выбора блока памяти связаны в системе с различными точками. Например, выход 1, на котором образуется сигнал выбора ПЗУ₀, связан с выводом 1 схемы IC_{16} . Необходимо убедиться, что логическое состояние вывода 1 схемы IC_{16} совпадает с логическим состоянием вывода 1 схемы IC_{11} .

11б. Проверить все линии выбора блоков памяти и их оконечные точки на правильность соединения и функционирования.

12. После завершения шага 11 можно сделать вывод о правильности соединения и отсутствии перегрузок всех линий адресной шины.

5.9. Проверка шины управления

Теперь можно приступить к проверке правильности функционирования шины управления. Для осуществления этого можно выполнить следующую процедуру. Отметим, что эту процедуру

нужно рассматривать как базовую; ее можно модифицировать с целью приспособления к каким-либо специфическим условиям. Процедура проверки шины управления состоит из следующих шагов.

1. Установить тумблеры управления на устройстве тестирования статическими сигналами в состояние логической 1.

2. Если все тумблеры управления установлены в состояние логической 1, то и на каждой линии шины управления в системе должно быть состояние логической 1. При этом ни одна линия управления не будет активной.

Сигналы управления микропроцессора Z80

Функция

$\overline{\text{IORQ}}$	$\overline{\text{MREQ}}$	$\overline{\text{RD}}$	$\overline{\text{WR}}$	
0	1	1	0	$\overline{\text{IOW}}$
0	1	0	1	$\overline{\text{IOR}}$
1	0	1	0	$\overline{\text{MEMW}}$
1	0	0	1	$\overline{\text{MEMR}}$

Рис. 5.11. Сигналы управления и соответствующие им функции системы.

3. Теперь тумблеры управления на устройстве тестирования устанавливаются в положения, соответствующие определенным функциям управления. Таких функций всего 4; эти функции и значения соответствующих сигналов управления представлены на рис. 5.11.

4. Установить с помощью тумблеров управления на устройстве тестирования комбинации, определяющие функции управления, и убедиться в том, что на линиях шины управления появляются правильные сигналы управления. Если на выходах шины управления обнаруживаются неправильные логические состояния, то нужно проверить корректность состояний на всех входах.

5. Если сигналы на входах шины управления правильны, однако на выходах шины управления устанавливаются ошибочные логические состояния, то, может быть, выходы оказались перегруженными. Снова целесообразно использовать стандартные методы отыскания неисправностей в цифровых схемах с целью определения причины обнаруженной ошибки в работе схемы.

6. Выходы шины управления связаны с несколькими точками системы. Необходимо убедиться в правильности установления логических состояний во всех точках системы, в которые параллельно подается сигнал управления.

7а. Например, сигнал шины управления $\overline{\text{BMEMR}}$ подается на входы различных логических схем. Одной из точек, в которую поступает этот сигнал, является вывод 2 схемы IC₁₂ (см. рис. 5.8а). Установим тумблеры управления на устройстве те-

стирования в состоянии, соответствующие сигналу чтения из памяти. Выполнив это, проверим вывод 2 схемы IC₁₂, чтобы убедиться в правильности реакции в этой точке на подачу сигнала ВМЕМР.

76. Выполнить аналогичные проверки в каждой точке системы, соединенной с выходом шины управления.

8. Проверка функционирования шины управления завершается после проведения контроля правильности действия всех сигналов управления, передаваемых по шине управления.

5.10. Проверка правильности подачи сигналов выбора кристаллов и сигналов разрешения записи в память

Теперь требуется проверить, что каждый модуль ПЗУ и ОЗУ правильно воспринимает адреса, поступающие по шине адреса, и правильно реагирует на сигналы управления, подаваемые по шине управления. Сначала проверим правильность подачи сигнала выбора кристалла ПЗУ0. Как показано на схеме, представленной на рис. 5.8а, вывод 20 схемы IC₁₄, представляющей собой ППЗУ типа 2708, является входом выбора кристалла. Этот вывод связан с выводом 3 схемы IC₁₆ 74LS32. Когда на выводе 3 — на выходе схемы IC₁₆ — устанавливается состояние логического 0, выходы данных ПЗУ0 (ППЗУ 2708) связываются с шиной данных.

После того как на дешифратор шины адреса будет подан правильный адрес ПЗУ0 и по шине управления поступит сигнал ВМЕМР, ПЗУ0 типа 2708 будет переведено в рабочее состояние. Это обеспечивается сигналом, который формируется на выходе схемы ИЛИ, и на рис. 5.8а обозначается IC₁₆. Когда и сигнал выбора памяти SEL₀, поступающий на вывод 1 схемы IC₁₆, и управляющий сигнал ВМЕМР, поступающий на вывод 2 той же самой схемы, имеют логические значения 0, на выводе 20 схемы IC₁₄ установится состояние логического 0.

Чтобы сформировать сигнал выбора ПЗУ0, необходимо все адресные тумблеры A₁₀—A₁₃ на устройстве тестирования установить в состояние логического 0. Тумблеры управления нужно установить в положения, соответствующие сигналу чтения из памяти, т. е. должны быть установлены следующие значения сигналов управления:

$$\overline{MREQ}=0, \quad \overline{RD}=0, \quad \overline{IORQ}=2, \quad \overline{WR}=1.$$

Отметим также, что, поскольку сигнал $\overline{RD}=0$, светонизлучающие диоды, находящиеся в устройстве тестирования, будут индцировать данные, поступившие на шину данных.

Если ПЗУ подключить к системе, светоизлучающие диоды будут отображать данные, хранимые в ПЗУ по адресу, определяемому положениями тумблеров A_0 — A_9 . Но если ПЗУ отключить от системы, то появится возможность выполнять некоторые интересные проверки.

Когда ПЗУ удалено из системы, то на всех светоизлучающих диодах устройства тестирования будет наблюдаться свечение. И если теперь с помощью куска провода поочередно заземлять выводы панели, на которой установлено ПЗУ, обнаружим, что светоизлучающий диод, соответствующий проверяемому разряду шины данных, перестает светиться. Этот способ дает возможность убедиться в корректности соединения всех линий шины данных с выводами ПЗУ. Например, после выполнения установки сигналов на шине адреса и на шине управления и после соединения с помощью куска провода вывода 9 панели для установки ПЗУ0 с «землей» мы должны увидеть, что светоизлучающий диод D_0 перестал светиться. Повторим выполнение процедуры для выводов 9, 10, 11, 13, 14, 15, 16 и 17. Таким способом убеждаемся в правильном соединении всех рассматриваемых выводов данных. Кроме того, мы установили, что при определенных состояниях линий шины адреса и шины управления выбор ПЗУ осуществляется правильно.

Рассмотренную процедуру необходимо выполнить для всех кодов выбора ПЗУ и для всех кодов выбора ОЗУ. При проверке ОЗУ также требуется подавать сигнал выбора кристалла в режиме чтения.

Правильность подачи сигнала разрешения записи для ОЗУ контролируется процедурой, которая подобна процедуре контроля ПЗУ. Основное отличие состоит в том, что по шине управления будет подаваться сигнал BMEMW, а не сигнал BMEMR.

Теперь рассмотрим, каким образом проверяется правильность дешифрирования всех сигналов выбора кристалла и сигналов разрешения записи в память. Рассмотрим также способ определения правильности физического соединения выходов данных ОЗУ и ПЗУ с линиями данных микропроцессора.

Отметим, что с помощью устройства тестирования статическими сигналами можно записывать данные ОЗУ и читать данные из ОЗУ. Указанные действия можно выполнить с помощью следующей процедуры:

1. Установить с помощью адресных тумблеров некоторый допустимый адрес ОЗУ.
2. С помощью тумблеров D_0 — D_7 на устройстве тестирования набрать код, определяющий данные, которые нужно записать по указанному адресу.
3. Установить тумблер, соответствующий сигналу управления MREQ, в положение 0.

4. Установить тумблер, соответствующий сигналу управления $\overline{WR}$, в положение 0. При этом активизируется линия подачи сигнала разрешения записи в память.

5. Переключить тумблер, соответствующий сигналу управления $\overline{WR}$ в положение 1. Тем самым снимается сигнал разрешения записи в память.

Данные помещены в память. Операция записи данных в ОЗУ выполнена точно так, как ее выполнил бы микропроцессор Z80, работая по программе. Теперь, чтобы определить правильность выполнения операции записи, прочитаем данные, находящиеся в ОЗУ по заданному адресу. Для этого необходимо:

6. Установить все тумблеры на устройстве тестирования, определяющие подаваемые в систему данные, в состояние 0.

7. Установить тумблер, определяющий значение сигнала $\overline{RD}$, в положение 0. После подачи этого сигнала будет активизирована линия выбора кристалла ОЗУ.

8. Светоизлучающие диоды устройства тестирования будут индцинировать данные, которые были записаны в память при выполнении шагов 1—5. Отметим, что операция чтения данных из ОЗУ с помощью устройства тестирования выполняется так же, как это делается при нормальной работе микропроцессорной системы Z80.

Рассмотренные приемы записи данных в ОЗУ и чтения данных из ОЗУ окажутся полезными при выявлении дефектов в ячейках или в кристалле ОЗУ. Так как мы можем подать на шину адреса любой адрес и действительно выполнить запись данных в ОЗУ, а затем прочитать данные из ОЗУ, проверка подозреваемых ячеек памяти окажется простым делом.

5.11. Запись и чтение данных из устройств ввода-вывода

Операции ввода-вывода для некоторых внешних устройств выполняются так же, как и операция записи и операция чтения данных из памяти. Для иллюстрации этого рассмотрим порядок записи данных в системный индикатор, схема которого представлена на рис. 5.8в.

Порты выхода в рассматриваемой схеме имеют коды F0, F1 и F2. Каждому указанному порту соответствует пара разрядов отображаемого числа — младшие, средние и старшие разряды соответственно. Принцип работы этого индикатора подробно дан в гл. 4. Записывая данные на индикатор, можно проверить правильность его функционирования. Если при выводе информации на индикатор обнаруживается ошибка, то можно приостановить выдачу информации на индикатор и «за-

морозить» систему в некотором состоянии. Когда система находится в статическом режиме, легко установить, какая схема дешифрирования в устройстве вывода неисправна.

Будем записывать на системный индикатор следующие числа: 1, 2, 3, 4, 5 и 6. При таком порядке записи цифра 1 будет записана в старший разряд, а цифра 6 — в младший разряд индикатора. Запись выполняется в соответствии со следующей процедурой:

1. Установить с помощью адресных тумблеров адрес старших двух разрядов, т. е. адрес $00F2_{16}$.

2. Установить с помощью тумблеров, используемых для определения вводимых данных, число 12_{16} . Напомним, что при записи данных на индикатор одновременно записываются две цифры. Разряды $D_0—D_3$ шины данных используются для передачи одной цифры, а разряды $D_4—D_7$ — для передачи другой цифры.

3. Установить тумблер управления, соответствующий сигналу $\overline{IORQ}$, в положение 0. Этот сигнал переведет систему в режим запроса ввода-вывода.

4. Установить тумблер управления, соответствующий сигналу $\overline{WR}$, в положение 0. Этот сигнал разрешает выполнение записи данных на внешние устройства.

5. Установить тумблер управления, соответствующий сигналу $\overline{WR}$, в положение 1. Тем самым сигнал управления записью на устройстве ввода-вывода будет снят. Когда сигнал $\overline{WR}$ примет значение логической 1, в старшие два разряда индикатора будут записаны цифры, определяемые положениями тумблеров $D_0—D_7$ на устройстве тестирования.

Если данные не записываются на индикатор, то, нарушив обычную последовательность проверки, можно выполнить детальный пошаговый контроль соответствующих аппаратных средств. Например, можно установить адрес $00F2_{16}$ и проверить, перешла ли в активное состояние линия выбора порта $F2$. Когда сигнал $\overline{WR}$ примет логическое значение 0, на выводе 1 схемы IC_1 типа 74LS374, предназначенном для подачи синхронизирующих импульсов, установится уровень логического 0.

Короче говоря, при тестировании можно проверить правильность установления состояний в различных точках системы, обусловленных определенным типом передачи данных, генерируемых устройством тестирования статическими сигналами. Шаги 1—5 рассмотренной процедуры нужно повторить два раза для записи еще четырех цифр. Выполняя с помощью устройства тестирования описанный выше контроль, мы убедимся, что все схемы и все соединения в выходном индикаторе функционируют правильно.

5.12. Проверка функционирования схемы клавишного пульта с помощью устройства тестирования

Теперь обсудим методы быстрого определения правильности функционирования клавишного пульта. Рассматриваемая процедура позволит нам убедиться в том, что все логические схемы работоспособны и все клавиши действуют нормально.

Для установки в состояние логического 0 одной из горизонтальных входных линий R_0 — R_4 клавишного пульта необходимо выполнить операцию ЗАПИСЬ. Затем должна быть выполнена операция ЧТЕНИЕ. После этого следует поочередно нажимать клавиши единственного на данный момент активного ряда. При этом будем следить за состояниями светоизлучающих диодов устройства тестирования. Светоизлучающий диод, соответствующий нажатой клавише, а значит, и определенной вертикальной линии матрицы клавиатуры, перестанет светиться. Мы описали идею построения процедуры проверки клавишного пульта. Полная же процедура будет состоять из следующих шагов:

1. Установить с помощью адресных тумблеров устройства тестирования код 00FE. Этот адрес будет использоваться и при выполнении операции ЗАПИСЬ, и при выполнении операции ЧТЕНИЕ.

2. Установить тумблер управления, соответствующий сигналу $\overline{IORQ}$, в положение 0.

3. Набрать с помощью тумблеров D_0 — D_7 код FE. В соответствии с этим кодом в активное состояние будет переведена линия R_0 .

4. Установить тумблер управления, соответствующий управляющему сигналу $\overline{WR}$, в положение 0.

5. Установить тумблер управления, соответствующий сигналу $\overline{WR}$, в положение 1. Теперь фиксатор, выходы которого соединены с горизонтальными линиями матрицы клавиатуры, будет содержать корректные данные.

6. Переключить тумблер соответствующий сигналу управления $\overline{RD}$, в положение 0.

6а. Все светоизлучающие диоды должны светиться, так как в данный момент нет ни одной нажатой клавиши. Заметим, что не следует принимать во внимание диоды D_5 , D_6 и D_7 , так как в клавиатуре эти разряды не используются. В программе, описанной в гл. 4, было предусмотрено маскирование этих разрядов.

7. Теперь надо поочередно нажимать клавиши первого ряда клавиатуры. Эти клавиши имеют следующие обозначения: R_0C_0 , R_0C_1 , R_0C_2 , R_0C_3 , R_0C_4 . Пока клавиша нажата, убеждаемся в том, что соответствующий светоизлучающий диод перестает светиться. Например, при нажатии клавиши R_0C_0 должен «погаснуть» светоизлучающий диод D_0 .

7а. Если вопреки ожиданиям диод продолжает светиться, то для отыскания неисправности следует использовать стандартные методы поиска неисправностей в цифровых схемах.

8. Шаги 3—7 надо выполнить для каждой горизонтальной линии клавиатуры. При этом каждый раз надо вводить данные, определяющие активность следующей горизонтальной линии клавиатуры.

9. После проверки клавиш некоторого ряда следует переключить тумблер управления, соответствующий сигналу $\overline{RD}$, в положение 1.

Эта процедура проверки функционирования клавишного пульта очень эффективна. После ее выполнения можно быть уверенным в правильности работы клавишного пульта системы. Если при проверке клавиатуры обнаруживается неисправность, то ее локализация в системе проводится рассмотренными нами методами тестирования статическими сигналами.

5.13. Выводы

Мы рассмотрели метод полной отладки аппаратных средств микропроцессорной системы Z80. Он получил название «метод тестирования статическими сигналами». Были представлены методики проверки всех системных шин, памяти и устройств ввода-вывода.

Применение метода тестирования статическими сигналами не ограничено новыми системами, которые еще не работали. Он может эффективно использоваться для поиска неисправностей в системах, которые были в эксплуатации и отказали. Метод тестирования статическими сигналами также можно использовать для отладки интерфейса новых устройств ввода-вывода, которые могли быть спроектированы в дополнение к уже существующей системе.

Можно предположить, что, поскольку метод статического тестирования проверяет систему только в статическом режиме, нельзя гарантировать правильность работы системы при ее номинальной скорости. Было установлено экспериментально, что системы, выдержавшие статическое тестирование, с большой вероятностью будут нормально работать и при номинальной системной скорости. Этот вывод прежде всего относится к системам, которые уже работали, но по каким-либо причинам отказали.

В новых проектах микропроцессорных систем отказы могут возникать из-за разнообразных причин, в частности из-за шумовых помех, источниками которых могут быть линии питания. В таких случаях метод тестирования статическими сигналами не пригоден, и поэтому приходится применять другие методы поиска неисправностей.

Можно с уверенностью говорить, что если при тестировании в системе обнаруживаются неисправности, то она определенно не будет работать при номинальной скорости. Устройство тестирования статическими сигналами особенно полезно для определения правильности работы логических схем системы. Оно позволяет «заморозить» систему в некоторый момент времени, что облегчает выполнение поиска некоторых неисправностей. В настоящей главе приведен достаточный материал для тех, кто только начал применять устройство тестирования статическими сигналами. По мере приобретения опыта использования устройства тестирования статическими сигналами оно будет становиться все более полезным инструментом, пригодным для решения многих проблем поиска неисправностей в аппаратных средствах микропроцессорных систем.

Глава 6

ПРЕРЫВАНИЯ, РЕЖИМ ОЖИДАНИЯ И РЕЖИМ ПРЯМОГО ДОСТУПА К ПАМЯТИ В МИКРОПРОЦЕССОРАХ 8080, 8085, 6800 и Z80

В настоящей главе рассматриваются вопросы организации прерываний, режима ожидания и режима прямого доступа к памяти в микропроцессорах 8080, 8085, 6800 и Z80. Сначала обсудим каждый из указанных вопросов, а затем рассмотрим методы реализации аппаратных средств, применяемых для обеспечения этих режимов в обсуждаемых микропроцессорах. Будем также рассматривать некоторые специальные команды и принципы построения программного обеспечения, используемые для осуществления трех указанных режимов. Изложение вопросов программного и аппаратного обеспечения прерываний, режима ожидания и режима прямого доступа к памяти будет носить основополагающий характер. Разобравшись, как реализуются эти режимы в простых микропроцессорных системах, будет легко адаптировать их к микропроцессорным системам с более сложной архитектурой.

По каждому из трех рассматриваемых режимов изложение будет проводиться в следующей последовательности. Мы будем описывать основные идеи программного и аппаратного обеспечения рассматриваемых режимов и демонстрировать примеры их реализации в распространенных типах микропроцессорных систем. Затем будем рассматривать особенности реализации указанных режимов в каждом из четырех обсуждаемых нами микропроцессоров. Принципы реализации сравниваются между собой. Изучив материалы настоящей главы, вы сможете использовать микропроцессорные системы, в архитектуре которых реализованы рассмотренные здесь режимы.

6.1. Основные представления о прерываниях

Сначала дадим общие представления о том, что такое прерывание. В рассматриваемых нами системах микропроцессор действовал как системный контроллер, т. е. он всегда выполнял команды программы в том порядке, в котором они были расположены в памяти. Это значит, что никакое вмешательство пользователя не могло воздействовать на ход выполнения программы, по которой уже начал работать процессор. Система прерываний позволяет изменять ход выполнения программы на ос-

новании сигналов, поступающих в микропроцессор. Рис. 6.1 дает наглядное представление о возникновении прерывания программы. Рис. 6.1, *а* соответствует случаю непрерывного последовательного выполнения четырех шагов программы. Рис. 6.1, *б* показывает, что первые два шага программы выполняются так же, как и при непрерывном последовательном ее выполнении.

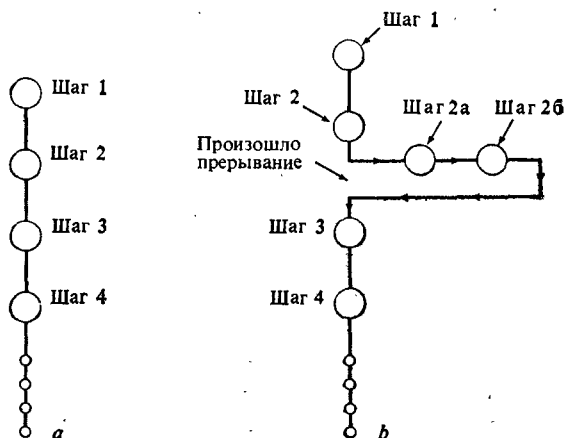


Рис. 6.1. Схема последовательного выполнения четырех шагов программы (*а*); схема, демонстрирующая появление запроса на прерывание после выполнения 1-го и 2-го шагов программы (*б*). Вследствие прерывания последовательное выполнение программы прекращается до тех пор, пока не выполнятся шаги 2а и 2б. После выполнения этих шагов продолжается нормальное выполнение прерванной программы.

Затем в микропроцессор поступает сигнал прерывания. После наступления прерывания выполнение программы прекращается и управление передается другой программной секции. Как только работа этой программной секции завершится, будет осуществлен переход к прерванной программе и продолжится ее нормальное выполнение.

На рис. 6.1, *б* показано, что, когда появляется сигнал прерывания, нормальный ход программы нарушается. Прерывание происходит между 2-м и 3-м шагами программы. Однако оно могло произойти и между другими шагами. Вообще, сигнал прерывания, поступающий в микропроцессор, является *асинхронным*, а это значит, что он может прийти на любом этапе работы программы. В рассматриваемом примере прерывание программы могло бы произойти и между шагами 3 и 4. Для системы не имеет значения, когда оно наступает, так как обработка прерываний всегда одинакова. В литературе по микропроцессорам такой принцип обработки прерываний называется *прерывание по вектору*.

В данном случае понятие *вектор* разнозначно понятию указателя. Этот вектор показывает микропроцессору, какой адрес нужно использовать, чтобы вызвать соответствующую программу обработки прерывания в момент его возникновения. Таким образом, понятие «вектор» здесь означает лишь только адрес памяти. Понятие «прерывание по вектору» означает, что в системе будут происходить прерывания, а вектор — это адрес начала программы обработки прерываний.

И если мы определим прерывание как вынужденный переход микропроцессора на выполнение определенной программы в момент поступления сигнала прерывания — это будет вполне оправдано. Значит, независимо от того, какую программную секцию выполняет микропроцессор, при наступлении прерывания он начнет выполнять другую программную секцию, после выполнения которой вернется к прерванной задаче.

В некоторых микропроцессорах и, в частности, в каждом из четырех микропроцессоров, рассматриваемых нами, используются два типа прерываний: 1) не маскируемые прерывания и 2) маскируемые прерывания. Если *прерывание не маскировано*, то микропроцессор будет реагировать на поступление соответствующего сигнала прерывания. Всякий раз, когда во время выполнения программы наступает немаскированное прерывание, микропроцессор должен среагировать на него.

Принцип *маскирования прерываний* состоит в том, что реакция микропроцессора на сигналы прерывания может быть разрешена или запрещена программным путем. Маскируемые прерывания могут эффективно отключаться программными средствами. Мы не будем обсуждать многочисленные вопросы, связанные с применением маскируемых и немаскируемых прерываний в микропроцессорных системах. Мы лишь покажем, как прерывания этих типов реализуются в каждом из четырех рассматриваемых микропроцессоров. Начнем с изучения системы прерываний для микропроцессора 8080.

6.2. Прерывания в микропроцессоре 8080

Для осуществления прерываний могут использоваться различные аппаратные средства, в частности схема, показанная на рис. 6.2. С помощью этой схемы сигнал прерывания может быть подан на любой из рассматриваемых нами микропроцессоров. Рассмотрев эту схему, мы сможем понять, как каждый процессор реагирует на запрос прерывания, поступающий от внешних аппаратных средств. Микропроцессор определенным образом связывается со схемой, представленной на рис. 6.2. На данном рисунке эта связь показана стрелкой, идущей от микропроцессора на вход сброса триггера 74LS74.

Кратко изложим принцип действия схемы, представленной на рис. 6.2. Слева на рисунке изображена схема, устраняющая влияние эффекта «дребезжания» ключа. Она состоит из двух инверторов с открытым коллектором и однополюсного двухходового кнопочного переключателя.

Когда ключ S_1 находится в состоянии покоя (NC), его центральный вывод соединен с нормально замкнутым полюсом. Нор-

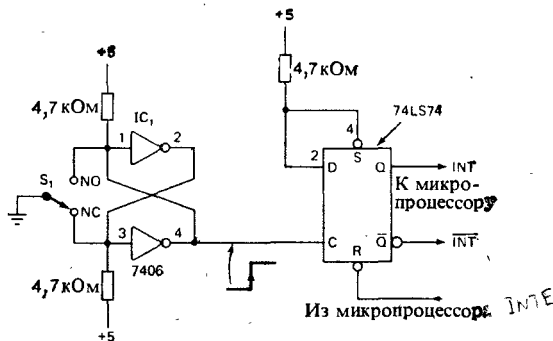


Рис. 6.2. Схема устройства, вырабатывающего сигнал прерывания микропроцессора.

мально замкнутый полюс соединяется с выводом 3 схемы IC_1 . На этом выводе будет уровень логического 0. Вывод 4 схемы IC_1 будет находиться в состоянии логической 1. На входе «синхроимпульсы» триггера 74LS74 также будет установлен уровень логической 1. Схема 74LS74 представляет собой триггер, запускаемый фронтом импульса. Когда на входе «синхроимпульсы» схемы 74LS74 происходит переход от уровня логического 0 к уровню логической 1, данные со входа D будут переданы на выходы Q и $\bar{Q}$. Исходя из рис. 6.2, можно предположить, что когда ключ находится в нормальном положении, на входе «синхронизация» триггера 74LS74 должен быть уровень логической 1. Если нажать кнопку S_1 , то центральный вывод S_1 соединится с нормально разомкнутым выводом переключателя. При этом установится уровень логического 0 на выводе 1 схемы IC_1 и уровень логической 1 на выводе 2 этой схемы. Следовательно, когда кнопка S_1 нажата, на входе «синхроимпульсы» схемы 74LS74 установится состояние логического 0. При этом состояние триггера не изменится.

Если теперь отпустить кнопку S_1 , то на входе «синхроимпульсы» схемы 74LS74 произойдет переход от состояния логического 0 к состоянию логической 1. Таким образом, посредством кратковременного нажатия кнопки S_1 обеспечивается передача данных со входа D на выходы Q и $\bar{Q}$. Временная диаграм-

ма этого процесса представлена на рис. 6.3. На вход D и вход установки триггера, которым в схеме 74LS74 соответствуют выходы 2 и 4, через резистор с сопротивлением 4,7 кОм подается напряжение +5 В. Таким образом на входах D и S триггера 74LS74 всегда поддерживается уровень логической 1. Данные передаются во время нарастания переднего фронта импульса на входе «синхроимпульсы»: выход Q всегда переходит в состояние логической 1, а выход $\bar{Q}$ — в состояние логического 0. Так формируется сигнал прерывания микропроцессора.

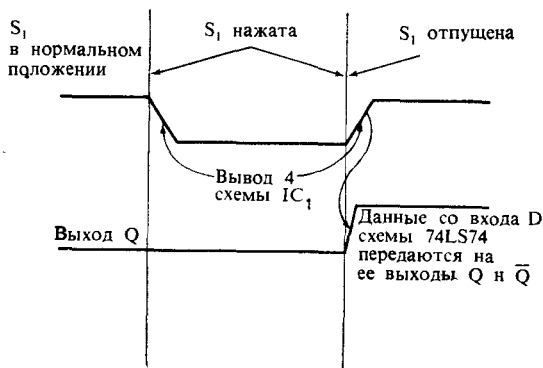


Рис. 6.3. Временная диаграмма сигнала на выходе Q, определяемого положением кнопки S_1 . Когда кнопка отпускается, на выходе Q устанавливается состояние логической 1, а на выходе $\bar{Q}$ — состояние логического 0.

После поступления сигнала прерывания микропроцессор должен в ответ послать сигнал, имеющий логическое значение 0, на вход «сброс» триггера 74LS74. Это обеспечит переход выхода Q в состояние логического 0, а выхода $\bar{Q}$ — в состояние логической 1. Так микропроцессор выполняет «сброс» запроса прерывания, выработанного внешними аппаратными средствами.

Используя рассмотренную схему, можно формировать и асинхронно подавать на определенные входы микропроцессора сигналы прерывания. Микропроцессор может выполнять программу, а мы, нажимая в некоторый момент времени на кнопку S_1 , посылаем сигнал прерывания. Когда будут рассматриваться специфические вопросы обработки прерываний в микропроцессорах, мы снова обратимся к схеме, представленной на рис. 6.2, чтобы более подробно разобраться в том, почему для генерации запроса на прерывание необходимо использовать триггер. (Сейчас этот вопрос рассматривать преждевременно. В нем будет легче разобраться после подробного рассмотрения системы прерываний в микропроцессоре 8080.)

На рис. 6.4 показана схема подключения аппаратных средств, обеспечивающих формирование запроса на прерывание (см. рис. 6.2), к микропроцессору 8080. Этот интерфейс, очевидно, очень прост. Линия, по которой в микропроцессор поступает запрос на прерывание, имеет обозначение INT. Она подводится к выводу 14 микропроцессора 8080. Этот микропроцессор имеет специальный выход INTE (вывод 16), который используется для подачи сигнала на вход «Сброс» схемы 74LS74.

В процессе обработки прерывания, запрос на которое уже, предположим, поступил, можно выделить следующие два этапа:

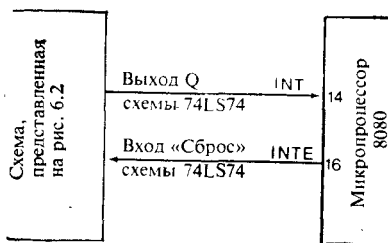


Рис. 6.4. Блочная схема подключения формирователя сигнала прерывания, схема которого дана на рис. 6.2, к микропроцессору 8080.

1. Вход INT (вывод 14 микропроцессора 8080) перешел в состояние логической 1. Это означает, что прерывание запрашивается внешними аппаратными средствами.

2. Микропроцессор 8080 принял запрос на прерывание, поступивший на вывод 14, на выходе INTE (вывод 16) будет установлен уровень логического 0.

Эти два события являются основными в рассматриваемом интерфейсе. Однако имеются и другие аппаратные средства, используемые в микропроцессоре 8080 для обработки прерываний. Пока мы только рассмотрели, как образуются запросы на прерывания и как выполняется «сброс» запроса на прерывание. Когда вывод 16 микропроцессора 8080 находится в состоянии логической 1, микропроцессор находится в режиме приема запроса на прерывание от внешнего оборудования. Если на выходе INTE (вывод 16) устанавливается состояние логического 0, микропроцессор игнорирует все запросы на прерывания, поступающие на вход INT (вывод 14). В микропроцессоре 8080 используются только маскируемые прерывания.

В системе команд микропроцессора 8080 есть две специальные команды: команда запрещения прерываний DI и команда разрешения прерываний EI. С помощью этих двух команд можно изменять состояние выхода INTE. Например, когда выполняется начальная установка микропроцессора 8080, выход INTE переводится в состояние логического 0. Это значит, что непосредственно после установки начального состояния микро-

процессора, он будет игнорировать все запросы на прерывание, поступающие от внешних аппаратных средств. Действительно, микропроцессор 8080 будет работать в режиме запрета прерываний. Единственный способ, который позволяет перевести выход INTE в состояние логической 1, состоит в применении команды разрешения прерываний EI. После выполнения команды разрешения прерывания выход INTE окажется в состоянии логической 1.

Если на выходе INTE состояние логической 1, то и на входе «сброс» триггера 74LS74 будет состояние логической 1. Тем самым допускается возможность передачи запроса прерывания со входа D на выход Q схемы 74LS74. Предположим теперь, что извне поступил запрос на прерывание. Когда запрос на прерывание воспринят, с помощью внутренних аппаратных средств микропроцессора выход INTE снова перейдет в состояние логического 0.

Что же произойдет, когда микропроцессор снова будет переведен в режим запрета прерываний? Единственный способ разрешить прерывания в микропроцессоре 8080 — это выполнить команду EI. Таким образом, выход INTE может быть переведен в состояние логического 0 либо с помощью аппаратных, либо с помощью программных средств микропроцессора 8080. Однако в состоянии логической 1 этот выход можно перевести только программно.

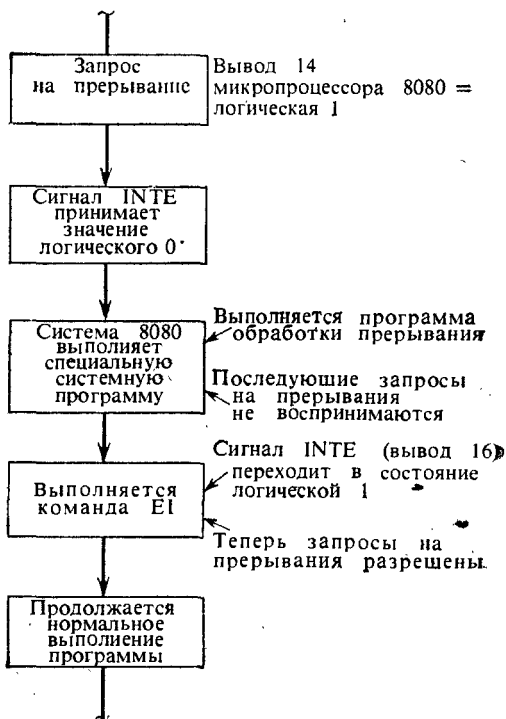
Указанная особенность требует специального рассмотрения. Отметим прежде всего, что программное обеспечение микропроцессора 8080 должно содержать специальную программу, назначение которой разрешать прерывания в некоторые точно определенные моменты времени. Это значит, что после выполнения программы прерывания возможность прерывания должна быть снова восстановлена. Действительно, во время выполнения программы прерывания микропроцессор 8080 не может воспринимать другие запросы на прерывания. Так будет до тех пор, пока выполнение этой программной секции не завершится. Блок-схема, отражающая последовательность событий, происходящих в системе после возникновения прерывания, представлена на рис. 6.5. Рассмотрим эту блок-схему.

Запрос на прерывание поступает на вывод 14 микропроцессора 8080. Микропроцессор устанавливает уровень логического 0 на выходе INTE (вывод 16). После этого система приступает к выполнению специальной программы. Пока не будем обсуждать, каким образом система получает начальный адрес этой программы. Допустим, что система каким-либо образом его получила и начала выполнение программы, во время которого на выходе INTE будет состояние логического 0. Следовательно, микропроцессор 8080 не будет воспринимать запросы на прерывания. Однако программа прерывания завершается командой

EI и на выходе INTE снова устанавливается уровень логической 1. Кроме того, прерванная программа продолжит работу. Подробное рассмотрение того, как возобновляется выполнение прерванной программы, будет сделано ниже.

Измененный вариант рассмотренной нами блок-схемы представлен на рис. 6.6. В новом варианте блок-схемы после пере-

Рис. 6.5. Блок-схема последовательности действий, выполняемых в системе 8080 после возникновения прерывания.



вода выхода INTE в состояние логического 0, осуществляемого аппаратными средствами, воспринявшим запрос на прерывание, выполняется команда разрешения прерывания. Когда выполнится команда EI, вывод INTE перейдет в состояние логической 1. Теперь микропроцессор 8080 может воспринимать новые запросы на прерывание.

Система начнет выполнять программу обработки прерывания. При этом она запросит память, необходимую для выполнения этой программы. Однако программа обработки прерывания может быть снова прервана тогда, когда появится другой запрос на прерывание. В функции программного обеспечения входит определение того, может ли прерываться процесс выполнения программы обработки прерывания или он должен завершиться до того, как начнут восприниматься последующие за-

просы на прерывания. Указанная особенность организации системы прерываний характерна для большинства микропроцессоров.

Теперь рассмотрим, как запрос на прерывание вводится в микропроцессор 8080 и каким образом микропроцессор реагирует на этот запрос. Сначала остановимся на вопросе организа-

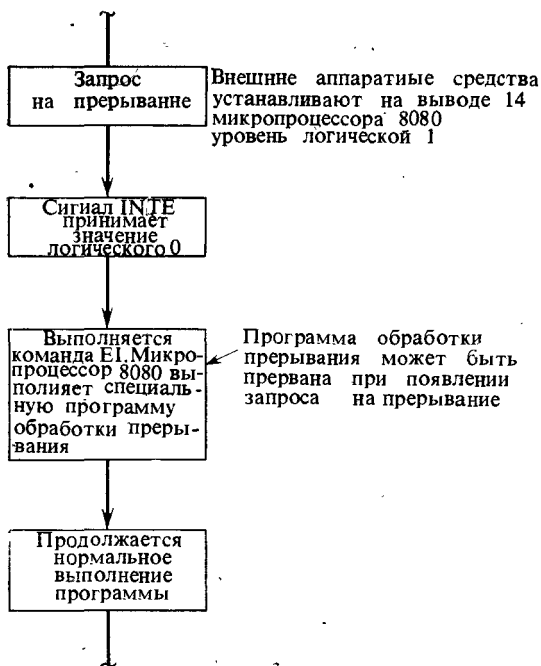


Рис. 6.6. Блок-схема последовательности действий, выполняемых в системе 8080 после появления запроса на прерывание в случае, когда допускается прерывание программы обработки прерывания.

ции обращения системы к программе обработки прерываний, которая вызывается после наступления прерывания. Внешние аппаратные средства должны в определенное время обеспечить ввод в микропроцессор адреса или вектора, определяющего точку входа в программу обработки наступившего прерывания.

К четырем известным нам сигналам управления (ЗАПИСЬ В ПАМЯТЬ, ЧТЕНИЕ ИЗ ПАМЯТИ, ЗАПИСЬ В УСТРОЙСТВО ВЫВОДА и ЧТЕНИЕ ИЗ УСТРОЙСТВА ВВОДА), передаваемым по шине управления, нужно добавить новый сигнал, который будет называться сигналом **ПОДТВЕРЖДЕНИЯ ПРЕРЫВАНИЯ**. Он будет иметь обозначение **INTA**. Когда подается сигнал управления **INTA** с логическим значением 0, внешние аппаратные средства системы должны поместить на шину дан-

ных вектор прерывания. Соответствующая последовательность действий представлена блок-схемой, приведенной на рис. 6.7.

Анализируя блок-схему, мы видим, что после поступления запроса на прерывание на выводе $\overline{INTA}$ устанавливается состояние логического 0, затем подается сигнал $\overline{INTA}$ с логическим значением, равным 0. Теперь с помощью определенных аппаратных средств должен обеспечиваться ввод вектора прерываний в

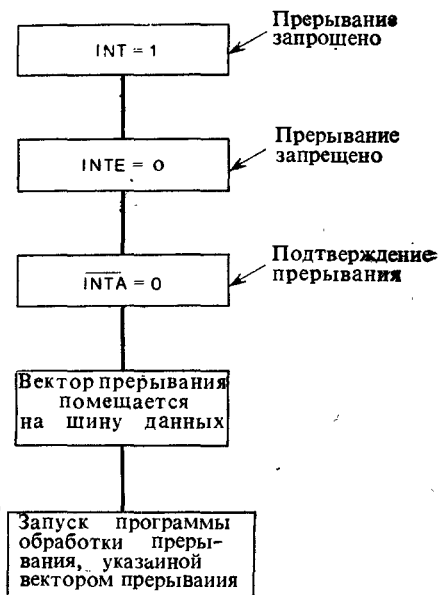


Рис. 6.7. Блок-схема последовательности действий, выполняемых внешними аппаратными средствами системы 8080, после поступления сигнала внешнего прерывания.

микропроцессор. В системе 8080 имеется восемь векторов прерываний, что позволяет системе начинать выполнение программы прерывания с одной из восьми определенных ячеек памяти. Векторы и соответствующие им адреса памяти представлены на рис. 6.8.

В системе 8080 векторы прерываний называют также векторами рестарта, а для их обозначения используется аббревиатура RST. Таким образом, векторы рестарта будут иметь следующие обозначения: RST_0 , RST_1 , RST_2 , RST_3 , RST_4 , RST_5 , RST_6 и RST_7 . При этом вектору RST_0 соответствует адрес 0, вектору RST_1 соответствует адрес 8 и т. д. Если, например, в микропроцессор вводится вектор RST_3 , то на шину данных поступит слово данных, соответствующее вводимому вектору.

На рис. 6.9 представлена схема, обеспечивающая подачу вектора рестарта на шину данных. Подачу вектора рестарта на шину данных называют также «вклиниванием» вектора рестарта. Видно, что эта операция подобна чтению данных из памяти

или чтению данных из устройства ввода. Единственное отличие состоит в том, что функция ввода данных в микропроцессор 8080 имеет другое назначение.

Реализация аппаратных средств для выполнения прерывания не вызовет трудности. Однако относительно одной из пяти перечисленных нами особенностей, связанной с выполнением ЦП ввода данных, находящихся на системной шине данных, заметим, что синхронизацию этого процесса осуществляет микропроцессор. При обработке прерываний микропроцессор освобождает нас от учета всех временных ограничений. Достаточно только декодировать определенные данные и в течение определенного периода времени поместить их на шину данных.

Вектор рестарта	Разряды слова данных								Адрес памяти
Имя	D_7				D_0				
RST ₀	1	1	0	0	0	1	1	1	0 ₁₆
RST ₁	1	1	0	0	1	1	1	1	8 ₁₆
RST ₂	1	1	0	1	0	1	1	1	10 ₁₆
RST ₃	1	1	0	1	1	1	1	1	18 ₁₆
RST ₄	1	1	1	0	0	1	1	1	20 ₁₆
RST ₅	1	1	1	0	1	1	1	1	28 ₁₆
RST ₆	1	1	1	1	0	1	1	1	30 ₁₆
RST ₇	1	1	1	1	1	1	1	1	38 ₁₆

Рис. 6.8. Соответствие между именами, разрядами слова данных и адресами памяти векторов рестарта.

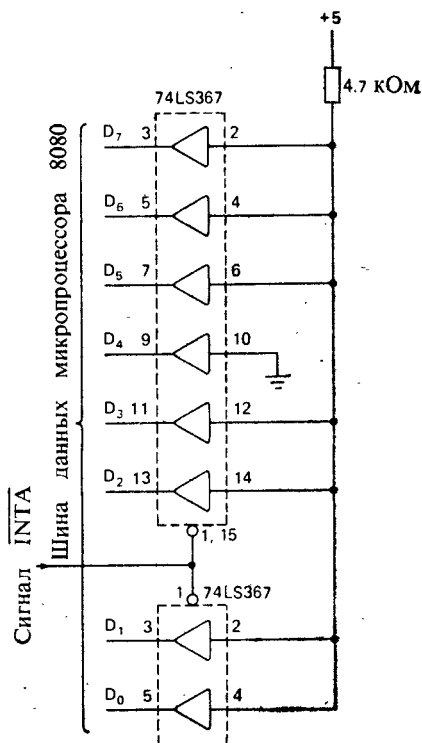
Теперь рассмотрим, как осуществляется ввод запроса на прерывание и как запускается особая программа обработки прерывания. Потом мы обсудим, каким образом после завершения программы обработки прерывания возобновляется выполнение прерванной программы. Для достижения указанной цели необходимо рассмотреть ту часть программного обеспечения системы 8080, которая используется в процессе обработки прерывания.

На рис. 6.10 представлена последовательность событий, происходящих в системе 8080 после поступления запроса на прерывание от внешних аппаратных средств системы. Мы видим, что, после того как произошло прерывание, на выводе INTE устанавливается уровень логического 0. Затем вырабатывается сигнал шины управления, и в это время вектор рестарта вводится в микропроцессор. Наконец, адрес возврата помещается в системный стек.

Подобная последовательность действий осуществляется, когда при выполнении некоторой программы реализуется вызов подпрограммы. Когда в системе 8080 происходит обращение к

подпрограмме, микропроцессор помещает в стек системы адрес возврата. При желании вернуться к основной программе после выполнения подпрограммы необходимо просто выполнить команду возврата (RET). При выполнении команды RET адрес возврата извлекается из стека, и стек снова может использоваться для хранения адресов возврата.

Рис. 6.9. Схема подачи вектора рестарта на шину данных за заданное время. Схема соответствует вектору рестарта RST 5.



Такая же временная последовательность событий происходит и при обработке прерываний. Микропроцессор помещает адрес возврата в системный стек. Когда после обработки прерывания возникнет необходимость возвращения к основной программе, в программе обработки прерывания должна быть выполнена команда возврата RET. При выполнении команды RET из стека будет выбран адрес, который был помещен туда перед началом выполнения программы обработки прерывания. Используя извлеченный из стека адрес, микропроцессор приступает к выполнению прерванной программы.

Рассмотрим теперь другие вопросы, связанные с обработкой прерываний в микропроцессоре 8080. После наступления прерывания микропроцессор 8080 переходит к выполнению специаль-

ной программы, находящейся в памяти по определенному адресу. Вспомним, что запрос на прерывание может поступить в произвольный момент времени. Это значит, что микропроцес-

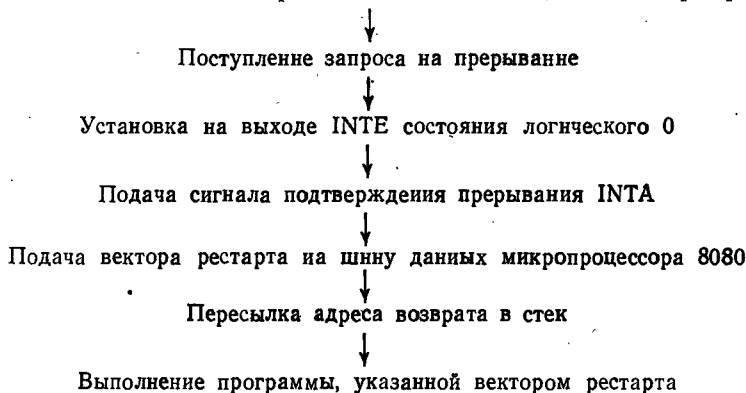


Рис. 6.10. Последовательность действий, выполняемых микропроцессором 8080.

сор должен обеспечить быстрое принятие решения, основываясь на состоянии некоторых внутренних флажков. Так как во время выполнения программы обработки прерываний содержимое этих флажков или регистров может быть разрушено, необхо-

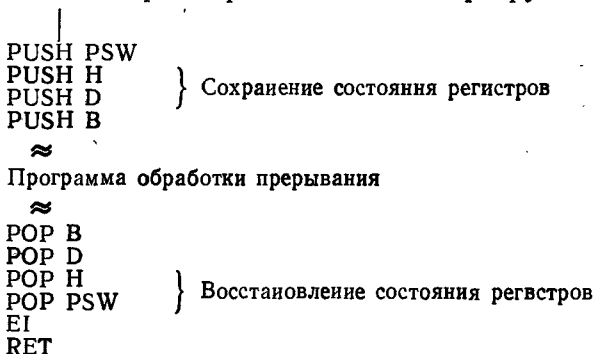


Рис. 6.11. Фрагмент программы, иллюстрирующий, как сохраняется и восстанавливается «состояние системы».

димо «сохранить» их состояния. Это можно осуществить посредством записи содержимого регистров в стек сразу после входа в программу обработки прерываний. А потом, непосредственно перед выходом из программы обработки прерываний, нужно будет восстановить их состояния. Рассмотренная идея иллюстрируется программой, представленной на рис. 6.11.

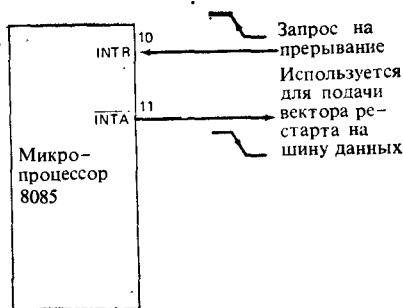
Подведем итоги обсуждения вопроса организации прерываний в микропроцессорной системе 8080. Запрос на прерывание

поступает от внешних аппаратных средств. Последующие запросы на прерывания могут быть разрешены или запрещены программно. Подачей сигнала шины управления INTA обеспечивается помещение на шину данных вектора прерывания. Вектор прерывания является адресом или указателем, определяющим некоторую точку входа в программу обработки прерываний. Прежде чем приступить к выполнению программы обработки прерывания, начало которой определяется вектором прерывания, в стек помещается адрес возврата. Последнее действие выполняется подобно тому, как это делается при обращении к подпрограмме. Для возвращения в прерванную программу последней в программе обработки прерываний должна выполняться команда RET.

6.3. Прерывания в микропроцессоре 8085

Прерывания в микропроцессоре 8085 могут осуществляться несколькими способами. Один из них полностью совпадает с рассмотренным нами способом организации прерываний в микропроцессоре 8080. Вход микропроцессора 8085, на который по-

Рис. 6.12. Схема, демонстрирующая выводы микропроцессора 8085, используемые для подачи запроса на прерывание и для выдачи сигнала подтверждения.



дается запрос на прерывание, имеет обозначение INTR. Этому входу соответствует вывод 10. Сигнал подтверждения прерывания INTA (вывод 11 микропроцессора 8085) фактически является сигналом разрешения подачи вектора рестарта на шину данных. Упомянутые выводы и соответствующие им сигналы показаны на рис. 6.12. На рисунке показано, что когда подается сигнал подтверждения прерывания INTA, т. е. когда на выводе 11 уровень напряжения понижается, вектор рестарта подается на шину данных и микропроцессор начинает выполнять программу обработки прерывания, соответствующую полученному вектору рестарта. В данном случае векторам рестарта поставлены в соответствие точно такие же адреса памяти, как и в рассмотренной нами системе прерываний для микропроцессора 8080 (см. рис. 6.8). В отличие от микропроцессора 8080 в микропроцессоре 8085 не предусмотрен сигнал INTE. Это означает,

что для выдачи сигнала сброса запроса на прерывание нужно использовать порт вывода. Согласно схеме, представленной на рис. 6.13, после поступления запроса на прерывание, вырабатываемого триггером типа D (см. схему на рис. 6.2), для сброса запроса на прерывание необходимо подать сигнал на вход «Сброс» триггера типа D. На рис. 6.13 показана линия сброса D-триггера, началом которой является выход порта вывода.



Рис. 6.13. Схема формирования сигнала сброса запроса на прерывание, в которой используется специальный порт вывода.

Чтобы снять запрос на прерывание, микропроцессор 8085 должен записать определенное слово данных в порт вывода.

В микропроцессоре 8085 также предусмотрен вход TRAP (вывод 6), предназначенный для подачи запросов на прерывание, которые не могут маскироваться. Это значит, что запросы на прерывание, подаваемые на вход TRAP, никогда не могут быть запрещены программно. Запросам на прерывание по входу TRAP присвоен самый высокий приоритет по отношению ко всем другим прерываниям. Если в один и тот же момент времени поступят хотя бы два запроса на прерывание, то запрос прерывания, поступивший на вход TRAP, будет иметь преимущество над всеми другими. При использовании прерываний по входу TRAP нет необходимости подавать вектор рестарта на шину данных. Этот вектор рестарта автоматически формируется и выдается микропроцессором 8085. Ему соответствует адрес рестарта 24_{16} . Для подачи запроса на прерывание на вход TRAP может быть использована схема, показанная на рис. 6.14. Отметим, что в данной схеме отсутствует триггер типа D. Вход TRAP является чувствительным как к фронту, так и к уровню сигнала. Сигнал на входе TRAP должен иметь уровень логической 1 в течение времени, которое требуется для его ввода в микропроцессор 8085. Однако микропроцессор устанавливает

появление сигнала на этом входе только тогда, когда на нем происходит переход к уровню логического 0 и обратно к уровню логической 1. Для этого кнопочный переключатель, обозначенный на рис. 6.14 через S_1 , нужно нажать и отпустить.

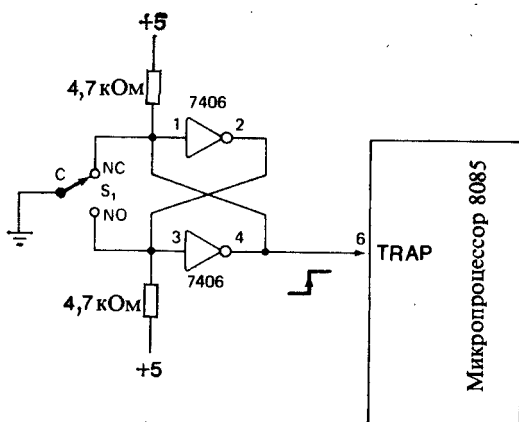


Рис. 6.14. Схема формирования сигнала прерывания TRAP в микропроцессоре 8085.

В микропроцессоре 8085 предусмотрено еще три входа для подачи запросов на прерывания. Эти входы имеют следующие обозначения: RST 7.5, RST 6.5, RST 5.5. Им соответствуют выходы 7, 8 и 9 микропроцессора 8085. При поступлении запросов на прерывание на указанные входы нет необходимости обеспечивать подачу адресов рестарта на шину данных — микропро-

Вход прерывания	Адрес памяти
5.5	$2C_{16}$
6.5	34_{16}
7.5	$3C_{16}$
Вектор рестарта	Адрес памяти
5	28_{16}
6	30_{16}
7	38_{16}

Рис. 6.15. Адреса памяти векторов рестарта. Отметим, что адрес вектора рестарта 6.5 расположен между адресами векторов рестарта RST 6 и RST 7.

процессор определяет их автоматически. Адреса для векторов рестарта даны на рис. 6.15. Отметим, что адреса векторов прерываний 5.5, 6.5 и 7.5 физически расположены между векторами рестарта.

В микропроцессоре 8085 сигналы рестарта 5.5 и 6.5 имеют такие же временные характеристики, как и сигнал прерывания

INTR. Сигнал RST 7.5 имеет некоторые отличия. Сигнал RST 7.5 является активным, нарастающим, чувствительным к фронту запросом на прерывание. Для осуществления запроса на прерывание требуется подать единственный импульс. Это означает, что запрос на прерывание RST 7.5 производится просто подачей импульса на вывод RST 7.5 микропроцессора 8085. Этот импульсный сигнал будет «запоминаться» до тех пор, пока запрос на прерывание система не обработает или не сбросит.



Рис. 6.16. Временная диаграмма сигнала запроса на прерывание RST 7.5.

Временная диаграмма, представленная на рис. 6.16, иллюстрирует некоторые особенности сигнала RST 7.5.

Наряду с отличиями в аппаратных средствах, используемых для организации прерываний в микропроцессорах 8080 и 8085, имеются также отличия и в соответствующем программном обеспечении.

Основное отличие обусловлено тем, что в системе команд микропроцессора 8085 есть команды «установка маски прерываний» (SIM). С помощью этой команды могут быть разблокированы только определенные линии запросов на прерывания. Это значит, что прерывания RST 5.5, RST 6.5 и RST 7.5 могут блокироваться так же, как запрос прерывания INTR. Единственным запросом на прерывание, который нельзя заблокировать программно, является запрос на прерывание TRAP. Прерывание по входу TRAP не может маскироваться и будет обрабатываться всегда.

```
MVI A, 18H
SIM
EI
```

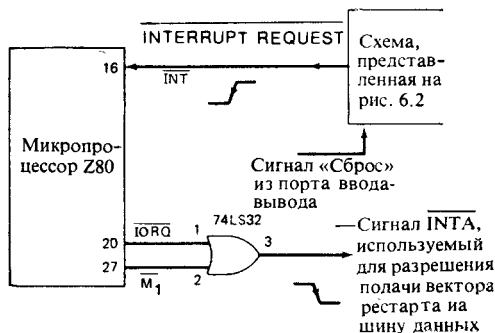
Рис. 6.17. Фрагмент программы для микропроцессора 8085, обеспечивающий установку маски прерываний. Код 18₁₆ определяет, что 4-му и 3-му разряду маски прерываний присваивается значение 1.

На рис. 6.17 представлен небольшой фрагмент программы для микропроцессора 8085, демонстрирующий применение команды установки маски прерываний SIM. Этот фрагмент обычно используют в начале программы и, кроме того, при обработке запросов на прерывания. Назначение разрядов маски прерываний описано в руководстве по программному обеспечению системы 8085.

6.4. Прерывания в микропроцессоре Z80

ЦП Z80 имеет два входа, предназначенные для приема запросов на прерывание: «запрос на прерывание» (INT, вывод 16) и «запрос на немаскируемое прерывание» (NMI, вывод 17). Сигнал запроса на прерывание INT является маскируемым, т. е. это прерывание может программным путем разрешаться или запрещаться. Немаскируемое прерывание будет восприниматься ЦП Z80 все время. Немаскируемое прерывание имеет вектор рестарта, равный 66_{16} . Это значит, что, когда произойдет

Рис. 6.18. Схема подачи запроса на прерывание и формирования сигнала подтверждения прерывания в микропроцессоре Z80.



немаскируемое прерывание, ЦП начнет выполнение той программы обработки прерывания, первая команда которой находится по адресу 66_{16} .

В микропроцессоре Z80 предусмотрено три различных способа обработки запросов на прерывания, поступающих на вход маскируемых прерываний, т. е. на вывод 16. Первый способ аналогичен реализованному в микропроцессоре 8080. Когда происходит прерывание, один из векторов рестарта RST_0 — RST_7 помещается на шину данных. Схема, иллюстрирующая описанные действия, приведена на рис. 6.18. Заметим, что для формирования сигнала $\overline{INTA}$ используются сигналы M_1 и $\overline{IORQ}$, вырабатываемые ЦП Z80. Сброс запроса на прерывание в данном случае выполняется так же, как и в микропроцессоре 8085. Для этого может быть использован особый порт вывода. Когда выполняется установка начального состояния микропроцессора Z80, сбрасывается и вход «запрос на прерывание» (вывод 16). Подтверждение поступления запроса на прерывание осуществляется способом, рассмотренным нами выше.

Второй способ реакции ЦП Z80 на прерывание состоит в следующем. Программным путем система может быть подготовлена к обработке прерывания так, что, когда аппаратные средства инициируют запрос на прерывание, автоматически бу-

дет установлен вектор рестарта 38₁₆. Этот способ обработки запроса на прерывание идентичен способу обработки запросов на прерывания RST 6.5 или RST 5.5 в микропроцессоре 8085, за исключением того, что в ЦП Z80 используется другой адрес рестарта. Подчеркнем, что выбор этого способа обработки прерываний осуществляется программно.

Рассмотрим теперь третий способ реакции микропроцессора Z80 на прерывания. В память системы можно записать таблицу

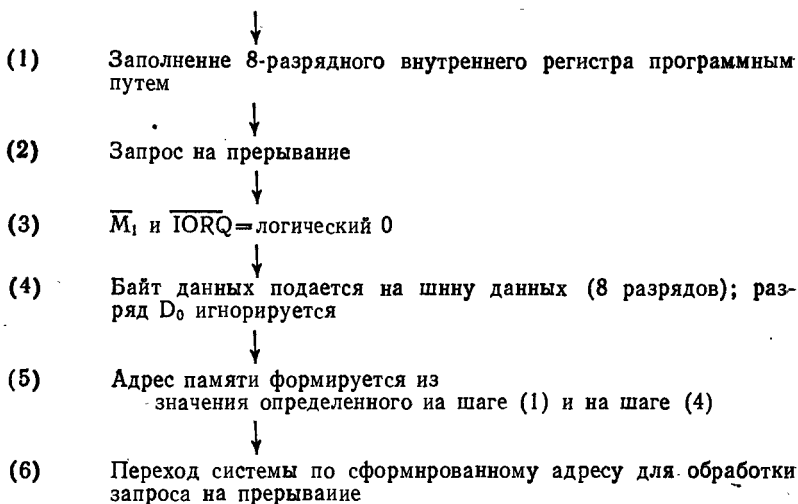


Рис. 6.19. Блок-схема последовательности действий, выполняемых в системе Z80 при возникновении запроса на прерывание. Восемь разрядов адреса подается на шину данных. Этот байт сцепляется с содержимым 8-разрядного внутреннего регистра, и таким образом формируется адрес программы обработки прерывания.

16-разрядных адресов памяти или указателей, которые соответствуют адресам программ обработки прерываний. Когда произойдет прерывание, внешние аппаратные средства должны поместить на системную шину данных младшие восемь разрядов определенного указателя, содержащегося в таблице. Старшие восемь разрядов этого указателя запоминаются во внутреннем регистре ЦП Z80. Объединяя указанные 8-разрядные коды, ЦП формирует слово. Это слово используется в качестве адреса памяти, по которому хранится указатель программы обработки прерывания. В зависимости от значений введенных младших разрядов формируемого слова система определит некоторый адрес 3FXX ячейки памяти. Сказанное выше иллюстрируется схемой на рис. 6.19. Предположим, что на шину данных поступил 8-разрядный код B6₁₆. Тогда будет сформирован указатель

3FB6, который и будет использован ЦП в качестве 16-разрядного адреса, указывающего местоположение программы обработки прерывания.

6.5. Прерывания в микропроцессоре 6800

Микропроцессор 6800 имеет два входа (IRQ, вывод 4 и NMI, вывод 6), предназначенные для ввода запросов на прерывания. IRQ является входом маскируемых запросов на прерывания.

Схема распределения памяти для векторов прерывания

Адрес	Комментарий
FFFF	Младшие 8 разрядов адреса
RESTART	
FFFE	Старшие 8 разрядов адреса
FFFD	Младшие 8 разрядов адреса
NMI	
FFFC	Старшие 8 разрядов адреса
FFFB	Младшие 8 разрядов адреса
SOFTWARE INT	
FFFA	Старшие 8 разрядов адреса
FFF9	Младшие 8 разрядов адреса
INT	
FFF8	Старшие 8 разрядов адреса

Рис. 6.20. Схема распределения памяти для векторов прерываний микропроцессора 6800.

а NMI предназначен для немаскируемых запросов на прерывания. Прерывание NMI не может быть запрещено программно. Когда поступает запрос на прерывание, микропроцессор обращается к особой ячейке памяти для получения адреса вектора рестарта, который будет использоваться для запуска программы обработки прерывания. Адреса рестарта приведены на рис. 6.20.

Аппаратные средства для выдачи немаскируемого запроса на прерывание микропроцессора 6800 представлены блочной схемой, изображенной на рис. 6.21. Они подобны тем, которые использовались для выработки запроса на прерывание в микропроцессорах 8085 и Z80. В микропроцессоре 6800 нет выхода INTE, который мог бы использоваться для выдачи сигнала сброса запроса на прерывание, поступившего от периферийного оборудования. Чтобы сбросить запрос на прерывание, необходимо записать в порт вывода определенную информацию. Теперь обратимся к рис. 6.20 и предположим, что ячейка памяти с адресом FFF8 содержит код 00_{16} , а ячейка с адресом FFF9 — код 35_{16} . Вектор рестарта или адрес, равный 0035_{16} , будет сформирован, когда от периферийного оборудования поступит запрос на прерывание.

Мы обсудили, как с помощью аппаратных средств вырабатываются запросы на прерывания в каждом из четырех рассматриваемых нами микропроцессоров. Способы выработки аппаратными средствами запросов на прерывания в рассматриваемых микропроцессорах одинаковы, поэтому легко разобраться в сходствах и отличиях соответствующих систем обработки прерываний. В реальных системах аппаратные средства для выдачи запросов на прерывания могут быть реализованы различными способами. При ознакомлении с какой-либо конкретной си-

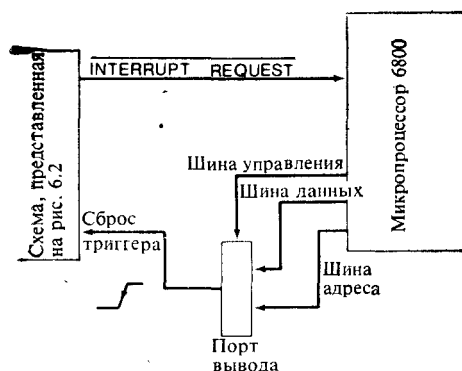


Рис. 6.21. Схема подключения формирователя запроса на прерывание, схема которого представлена на рис. 6.2, к микропроцессору 6800.

стемой прерываний прежде всего следует разобраться в аппаратных средствах микропроцессора, обеспечивающих разрешение и выполнение прерываний.

Надо помнить, что в микропроцессоре 8080 после наступления прерывания на шину данных необходимо подать вектор рестарта. Аналогичную операцию приходится выполнять и в одном из режимов обработки прерываний микропроцессора Z80. Точное время, затрачиваемое микропроцессором на выполнение прерывания, дается в перечне технических характеристик микропроцессора. Целью настоящего обсуждения было ознакомление с внешними аппаратными средствами, которые необходимы для формирования запросов и обработки прерываний.

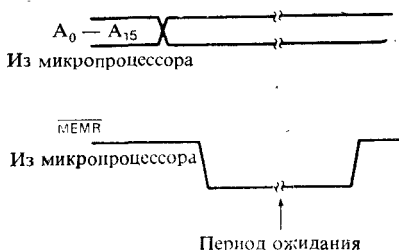
Большую помощь в изучении вопросов обработки прерываний оказывают имеющиеся в описании спецификаций микропроцессора временные диаграммы.

6.6. Способы реализации режима «ОЖИДАНИЕ»

Теперь рассмотрим различные методы обеспечения режима ОЖИДАНИЕ в оборудовании микропроцессорных систем. Будем использовать термин «ожидание» для обозначения некоторого периода времени, необходимого для выполнения операций чтения или записи в память или для выполнения операций

ввода-вывода. Когда время ответа (реакции) памяти или устройств ввода-вывода больше, чем время цикла команды, цикл команды должен быть увеличен. Период ожидания должен быть достаточным для срабатывания устройства памяти и устройств ввода-вывода. Сказанное иллюстрирует временная

Рис. 6.22. Временная диаграмма, демонстрирующая принцип генерации периода ожидания в микропроцессоре. Запрос памяти реализуется во время периода ожидания.



диаграмма, приведенная на рис. 6.22. В следующем разделе обсудим аппаратные средства, используемые для генерации состояния ожидания в каждом из четырех рассматриваемых микропроцессоров.

6.7. Перевод в состояние ожидания микропроцессоров 8080, 8085 и Z80

Аппаратные средства, предназначенные для перевода в состояние ожидания микропроцессоров 8080, 8085 и Z80, имеют большое сходство. Рассматриваемая нами схема применима для каждого из этих микропроцессоров; отличаются лишь способы подключения ее к выводам микропроцессора. Сначала обсудим, как осуществляется переход микропроцессора в состояние ожидания, а затем остановимся на рассмотрении интерфейса схем, обеспечивающих переход в режим ожидания, и микропроцессора.

Запрос на ОЖИДАНИЕ сигнал (WAITREQ) инициируется памятью системы или устройствами ввода-вывода. В рассматриваемом примере предполагается, что память системы имеет время доступа слишком большое, чтобы записывать или считывать данные на нормальной рабочей скорости микропроцессора. Поэтому, когда микропроцессор должен читать данные из памяти или записывать данные в память, время цикла ЦП должно быть увеличено. Это будет согласование времени доступа с медленнодействующей памятью. Таким образом, память может иметь секции, работающие с разным быстродействием. В частности, некоторые секции памяти будут обладать быстродействием, достаточным для работы на системной скорости. В этом случае потребуется «замедлять» микропроцессор только тогда, когда происходит обращение к медленнодействующей секции памяти.

Для выработки запроса на ожидание будут дешифровать-

ся старшие разряды адреса, передаваемого по адресной шине. Когда эти разряды адреса будут соответствовать низкоскоростной секции памяти, тогда произойдет выдача сигнала WAITREQ. Этот сигнал вырабатывается путем дешифрирования старших разрядов адреса с помощью комбинационных логических схем, принцип его формирования иллюстрируется блочной схемой, представленной на рис. 6.23. Мы видим, что адрес поступает на комбинационный логический блок, который дешифрирует адрес и вырабатывает сигнал WAITREQ, когда код адре-



Рис. 6.23. Схема образования сигнала запроса ожидания. При появлении определенной логической комбинации на шине адреса логическая схема вырабатывает сигнал запроса ожидания.

са соответствует медленнодействующей части памяти. Появление этого сигнала отмечается переходом напряжения на выходе комбинационной логической схемы от высокого уровня к низкому уровню.

Сигнал запроса на ожидание используется другими схемами, которые и вырабатывают сигнал, вызывающий переход ЦП в состояние ожидания.

Схема, предназначенная для генерации сигнала запроса на ожидание, представлена на рис. 6.24. Это обобщенная схема, которую путем расширения можно приспособить для выработки запросов ожидания любой желаемой длительности. Показанная схема будет обеспечивать задержку в работе системы на два периода ожидания. Мы обсудим в общих чертах эту схему и укажем, какие сигналы необходимы для функционирования ее с определенным микропроцессором. Отметим, что рассматриваемая схема может использоваться совместно с микропроцессорами 8080, 8085 и Z80. Ниже в настоящей главе будет представлена схема, предназначенная для работы с микропроцессором 6800.

Временная диаграмма, изображенная на рис. 6.25, дает представление о временных соотношениях сигналов, действующих в схеме, представленной на рис. 6.24. Сначала обратим внимание на последовательность тактовых импульсов. Все действия в рассматриваемой схеме происходят во время нарастания переднего фронта импульсов этой последовательности. Тактовые импульсы подаются на D-триггеры типа 7474. Сигнал запроса на ожидание подается на первый D-триггер рассматри-

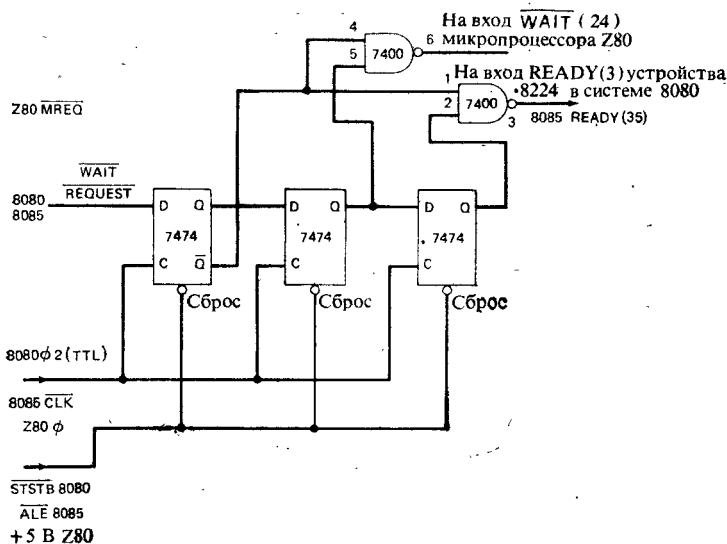


Рис. 6.24. Схема генерации состояния ожидания для микропроцессоров 8080, 8085 и Z80.

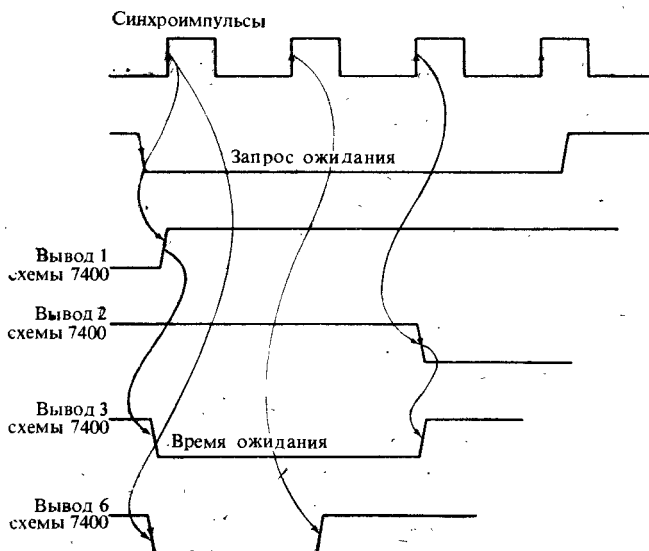


Рис. 6.25. Временная диаграмма сигналов, действующих в схеме, представленной на рис. 6.24.

ваемой схемы. Для схемы, показанной на рис. 6.23, этот сигнал является выходным. Предположим, что когда схема начинает работать, все ее триггеры на выходах Q имеют уровень логической 1. Справедливость этого предположения мы обсудим ниже.

Анализируя временную диаграмму, представленную на рис. 6.25, мы видим, что сигнал «запрос ожидания» на входе D первого триггера рассматриваемой схемы переходит к уровню логического 0, прежде чем сигнал генератора тактовых импульсов перейдет к уровню логической 1. Указанное временное соотношение между этими сигналами выполняется в каждом из трех обсуждаемых сейчас микропроцессоров; оно обеспечивается синхронизацией самих микропроцессоров. Когда синхронизирующий импульс изменится от уровня логического 0 до уровня логической 1, выход Q первого триггера типа D переходит в состояние логического 0, а выход Q этого же триггера переходит в состояние логической 1. При этом на обоих входах вентиля НЕ И, т. е. на выводах 1 и 2 схемы типа 7400, будет уровень логической 1. Следовательно, на выходе этого вентиля — вывод 3 схемы 7400 — установится уровень логического 0. Сигнал на выводе 3 схемы 7400 также представлен на временной диаграмме, изображенной на рис. 6.25. Отметим, что уровни рассмотренных сигналов остаются постоянными в течение времени существования второго импульса, показанного на временной диаграмме.

Во время нарастания переднего фронта третьего синхронизирующего импульса происходит следующее. На выходе Q третьего триггера типа D появится уровень логического 0; такой же уровень, очевидно, будет и на выходе 2 схемы 7400. Теперь вывод 3 схемы 7400 перейдет в состояние логической 1. Когда на этом выводе происходит переход к уровню логической 1, со входа микропроцессора снимается сигнал запроса ожидания. После снятия этого сигнала микропроцессор продолжит работу на нормальной системной скорости.

При переходе микропроцессора к нормальному режиму работы все триггеры рассматриваемой схемы устанавливаются в состояние 1 посредством подачи сигнала низкого уровня на входы «сброс». Сигнал низкого уровня подается на входы «сброс» в начале каждого нового машинного цикла микропроцессоров 8080 или 8085. Теперь схема готова к приему сигнала «запрос ожидания»; в зависимости от адреса, поступающего по адресной шине, на выходе схемы, представленной на рис. 6.24, установится состояние логического 0 или состояние логической 1.

Схема, изображенная на рис. 6.24, подключается к микропроцессорам 8080 и 8085 так, как это показано на рис. 6.26 и рис. 6.27 соответственно. При этом вывод 3 схемы 7400 (см. рис. 6.24) соединяется со входом $RDYIN$ генератора тактовых

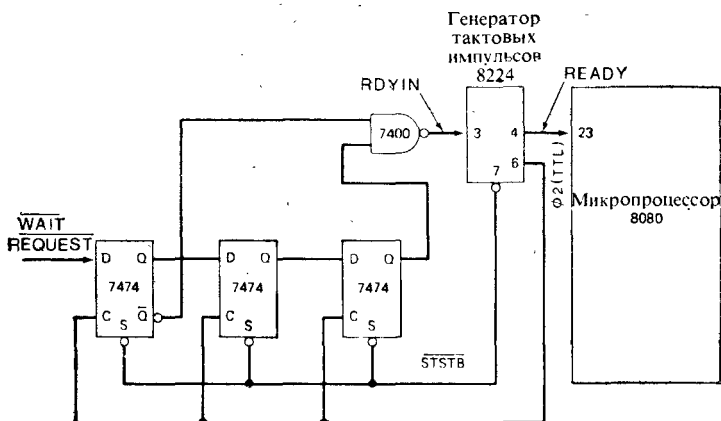


Рис. 6.26. Схема соединения микропроцессора 8080 со схемой, показанной на рис. 6.24.

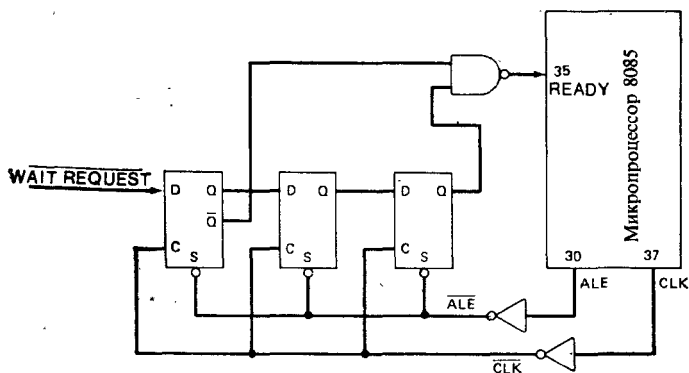


Рис. 6.27. Схема соединения микропроцессора 8085 со схемой, показанной на рис. 6.24.

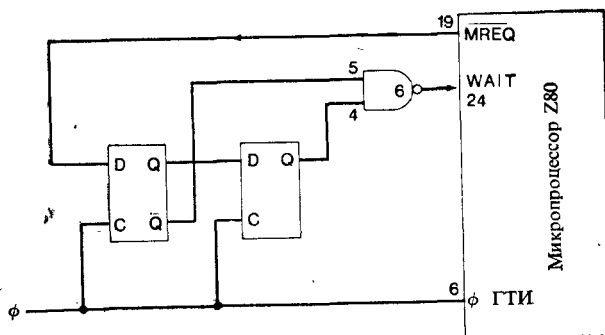


Рис. 6.28. Схема соединения микропроцессора Z80 со схемой, показанной на рис. 6.24.

импульсов 8224 микропроцессора 8080 или со входом READY микропроцессора 8085. На схему, показанную на рис. 6.24, синхронимпульсы поступают с выхода $\phi 2$ TTL генератора тактовых импульсов 8224 микропроцессора 8080. При использовании микропроцессора 8085 синхронимпульсы с выхода CLK сначала поступают на инвертор, а после инвертирования — на вход подключаемой к микропроцессору схемы (см. рис. 6.27). Для установки триггеров рассматриваемой схемы в состояние 1 используется сигнал $\overline{STSTB}$ — «строб состояния» — в микропроцессоре 8080 и сигнал $\overline{ALE}$ — «отпирание фиксатора адреса» — в микропроцессоре 8085.

Схема генерации сигнала «запрос ожидания» для микропроцессора Z80 представлена на рис. 6.28. Отметим, что эта схема работает точно так же, как и схема, рассмотренная нами выше. В данной схеме выход второго триггера соединяется с выводом 4 вентиля HE И. Эта схема обеспечивает формирование сигнала «запрос ожидания», длительность которого равна только одному периоду последовательности тактовых импульсов. Сигнал запроса ожидания на вход триггера типа D поступает с выхода микропроцессора MREQ. Для подачи синхронимпульсов используется линия, по которой поступают синхронимпульсы на вывод 6 микропроцессора Z80.

Перевод в состояние ожидания микропроцессора 6800

В отличие от микропроцессоров 8080, 8085 и Z80 микропроцессор 6800 не имеет входов для приема сигналов READY или WAIT. Способ «замедления» микропроцессора 6800 при обращении к памяти или к устройствам ввода-вывода носит название «растягивания синхронизирующих импульсов». Используя этот способ, фактически можно увеличивать ширину импульса последовательности $\phi 2$ до необходимой величины, определяемой временем доступа к памяти определенного типа. Согласно спецификациям, ширина импульса $\phi 2$ для микропроцессора 6800 не может превышать 4,5 мкс. Если увеличить ширину импульса $\phi 2$, то в микропроцессоре 6800 может произойти потеря информации. Это ограничение необходимо учитывать при растягивании импульса $\phi 2$.

В генераторе тактовых импульсов 6871, предназначенном для использования с микропроцессором 6800, предусмотрен вход, который посредством подачи на него сигнала позволяет замедлить или остановить импульс $\phi 2$, т. е. оставить его в данном состоянии, пока на входе действует сигнал запроса. Вход генератора 6871, используемый для этого, называется входом (линией) готовности памяти. Когда на линии готовности памяти низкий уровень напряжения, на входе $\phi 1$ микропроцессора бу-

дет состояние логического 0, а на входе $\phi 2$ — состояние логической 1. Для возобновления нормального режима работы генератора тактовых импульсов и, следовательно, всей системы необходимо подать на линию готовности памяти уровень логической 1.

Чтобы обратиться к медленнодействующей памяти, достаточно установить уровень логического 0 на линии готовности памяти. Схема, предназначенная для этого, представлена на рис. 6.29.

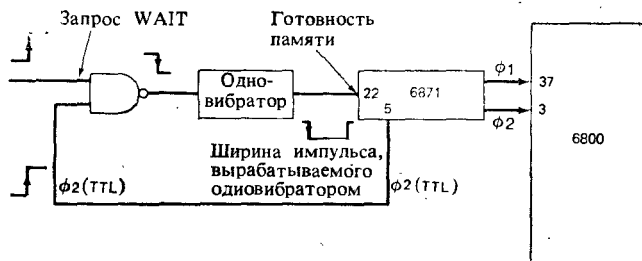


Рис. 6.29. Схема генерации состояния ожидания микропроцессора 6800.

Ждущий мультивибратор (одновибратор) запускается задним фронтом поступающего на его вход импульса. Сигнал, запускающий мультивибратор, поступит на его вход, когда и сигнал «запрос ожидания», и сигнал $\phi 2$ TTL, являющийся выходным сигналом генератора 6871, переходят к уровню логической 1. Когда указанные сигналы достигнут уровня логической 1, произойдет запуск ждущего мультивибратора и на его выходе появится уровень логического 0. Описанные изменения сигналов иллюстрируются временной диаграммой, представленной на рис. 6.30. Вместе с переходом к состоянию логического 0 выхода ждущего мультивибратора такое же состояние устанавливается и на линии готовности памяти. При этом последовательности тактовых импульсов «замораживаются» таким образом, что на входе $\phi 1$ микропроцессора устанавливается состояние логического 0, а на входе $\phi 2$ — состояние логической 1. Через определенное время на выходе мультивибратора, а значит, и на линии готовности памяти установится уровень логической 1. Теперь генератор тактовых импульсов продолжит работу в нормальном режиме.

Вспомним, что в микропроцессоре 6800 все действия на шине данных происходят при прохождении заднего фронта импульсов последовательности $\phi 2$. Это означает, что и передача данных на устройства вывода, и ввод данных в микропроцессор 6800 стробируются по заднему фронту импульсов фазы $\phi 2$. Поэтому и осуществляется задержка перехода импульса $\phi 2$ к уров-

нию логического 0 на время, задаваемое ждущим мультивибратором. Это время не должно превышать 4,5 мкс.

Кроме того, отметим, что, когда сигнал $\phi 2$ переходит к уровню логической 1, сигнал на линии готовности памяти перейдет к уровню логического 0, на входе $\phi 2$ уровень логической 1 будет сохраняться, пока ждущий мультивибратор находится в неустойчивом состоянии. После возвращения мультивибратора

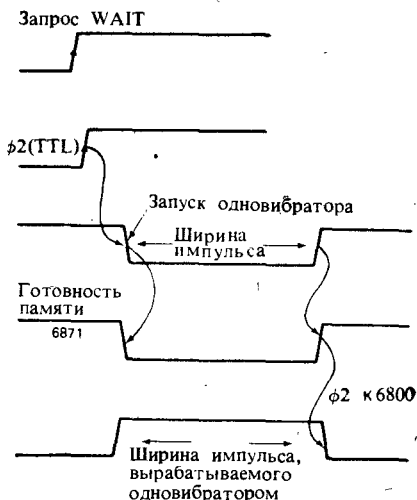


Рис. 6.30. Временная диаграмма сигналов, действующих в схеме генерации состояния ожидания микропроцессора 6800 (см. рис. 6.29).

в устойчивое состояние линия готовности памяти переходит в состояние логической 1 (см. рис. 6.30), а выход $\phi 2$ генератора тактовых импульсов 6871 — в состояние логического 0. Когда сигнал $\phi 2$ переходит к уровню логического 0, микропроцессор 6800 контролирует данные на шине данных или память принимает данные на свои входы.

Итак, мы рассмотрели способы перевода микропроцессоров 8080, 8085, Z80 и 6800 в состояние ожидания. Следует иметь в виду, что эти способы, конечно, не являются единственными. Представленные здесь схемы являются лишь примерами, которые позволяют нам увидеть, как с помощью аппаратных средств решается задача перевода микропроцессора в состояние ожидания. Хорошо понимая, что необходимо для перевода микропроцессора в состояние ожидания, легко и понять, как в системе работают аппаратные средства, обеспечивающие это состояние, и разработать схему для решения подобной частной проблемы.

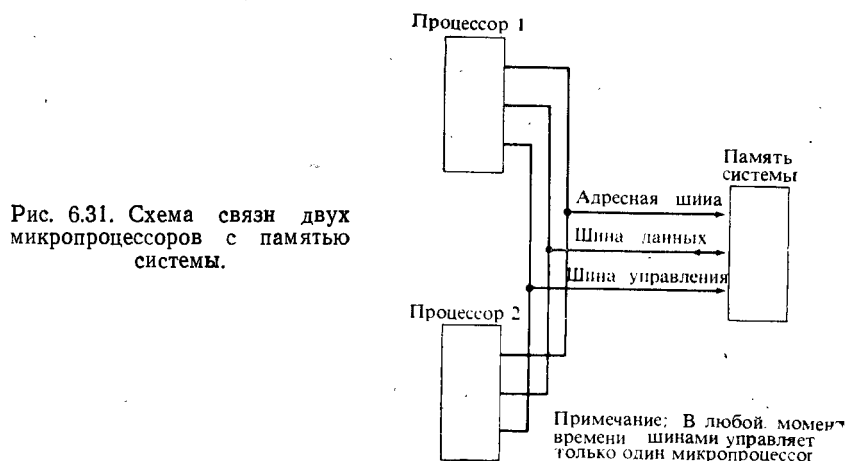
Кроме того, отметим, что существует другое решение этой задачи, которое может применяться по желанию. Это решение гораздо проще рассмотренных, однако не всегда может быть использовано. Оно заключается в уменьшении частоты генератора тактовых импульсов микропроцессорной системы. Если

микропроцессор работает слишком быстро и вследствие этого невозможно обеспечить интерфейс памяти стандартного типа, то можно просто уменьшить быстродействие микропроцессора и тем самым без использования какого-либо дополнительного оборудования обеспечить достаточное время для обращения к медленнодействующей памяти.

На первый взгляд это решение полностью неприемлемо. Однако требуемое время обращения к медленнодействующим устройствам обычно только на несколько сотен наносекунд больше, чем время, выделяемое для этого микропроцессором. И хотя уменьшение частоты генератора тактовых импульсов приводит к увеличению общего времени выполнения программы, при этом сокращается объем системных аппаратных средств. Естественно, в этом случае приходится все хорошо продумать. Однако отбрасывать это решение без предварительного анализа не следует никогда. Существует много приложений, в которых быстродействие микропроцессора не является определяющим. Тогда его уменьшение может оказаться выгодным — оно, быть может, приведет к желаемому уменьшению объема оборудования микропроцессорной системы.

6.9. Прямой доступ к памяти в микропроцессорах 8080, 8085, Z80 и 6800

Теперь обсудим, как при использовании рассматриваемых нами микропроцессоров реализуется прямой доступ к памяти (ПДП). В режиме прямого доступа к памяти не используется принцип



адресации памяти, обычно реализуемый в микропроцессорных системах. При ПДП адресацию будет выполнять какая-либо другая система или процессор; это значит, что какой-либо дру-

гой ЦП будет иметь доступ к линиям адреса памяти, линиям шины данных и к системной шине управления. При этих условиях микропроцессор, нормально связанный с памятью, должен передать управление всеми сигнальными линиями другому ЦП. Такая ситуация возникает, когда в некоторую систему входят по крайней мере два микропроцессора. Последние используют одно и то же пространство памяти. Простой пример такой системы представлен на рис. 6.31.

Рассмотрим по очереди, как каждый из представленных в этой книге микропроцессоров передает управление адресными линиями, линиями шины данных и линиями шины управления другому микропроцессору, которому необходим доступ к памяти, и покажем, какие выводы и аппаратные средства используются для обеспечения этого управления.

6.10. Прямой доступ к памяти в микропроцессоре 8080

Микропроцессор 8080 имеет вывод, носящий наименование **HOLD**, который позволяет использовать микропроцессор 8080 в режиме прямого доступа к памяти. В режиме нормального выполнения программы, т. е. тогда, когда режим ПДП не используется, вход **HOLD** находится в состоянии логического 0. При этом условии адресная шина, шина данных и шина управления управляются микропроцессором 8080. Когда другой микропроцессор или система требует использования этих трех шин, вход **HOLD** микропроцессора 8080 переводится в состояние логической 1. При этом происходит следующее:

- 1) адресные линии переходят в состояние высокого сопротивления;
- 2) линии управления переходят в состояние высокого сопротивления;
- 3) шина данных переходит в состояние высокого сопротивления.

Теперь другой процессор сможет осуществлять управление адресными линиями, линиями данных и линиями шины управления. В том случае, когда микропроцессор 8080 используется совместно с контроллером системы 8228, на запрос **HOLD** реагирует контроллер системы и переведет выходную шину и шину управления в состояние высокого сопротивления. Это достигается подачей сигнала на вход **HLDA** устройства 8228.

Выход **HLDA** — вывод 21 микропроцессора 8080 — используется для выдачи сигнала на внешнее оборудование. Этот сигнал является реакцией микропроцессора 8080 на запрос **HOLD**, поступающий на вывод 13. Последнее означает, что периферийное оборудование должно сначала подать запрос на ПДП посредством установления уровня логического 0 на выводе 13

микропроцессора 8080. Затем периферийное оборудование ожидает появления сигнала HLDA с логическим значением, равным 1. После установления на выходе HLDA состояния логической 1 микропроцессор 8080 перестает управлять всеми шинами системы. Теперь периферийное оборудование может производить прямой доступ к памяти. После завершения ПДП периферийное оборудование переводит вход HOLD микропроцессора

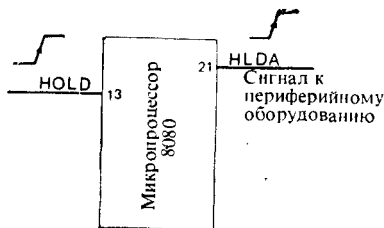


Рис. 6.32. Схема, демонстрирующая сигнал на выходе HLDA, который вырабатывается после поступления запроса HOLD на вывод 13 микропроцессора 8080.

сора 8080 в состояние логического 0. Используя выход HLDA и вход HOLD, периферийное оборудование может надежно выполнять прямой доступ к памяти (рис. 6.32).

Если же микропроцессор 8080 используется без контроллера 8228, то ПДП осуществляется следующим образом. Напомним, что адресная шина в микропроцессорной системе 8080 может иметь, но может и не иметь буфера. Если буфер имеется, то следует предусмотреть, чтобы он был с тремя устойчивыми состояниями. В рассмотренных нами прежде аппаратных средствах в качестве буфера адресных линий использовалось устройство 74LS367. Схема 74LS367 переводится в состояние высокого сопротивления посредством подачи сигнала уровня логической 1 на выводы 1 и 15 этой схемы. В данном случае таким сигналом будет HLDA, вырабатываемый микропроцессором 8080. Таким образом, появление на выходе HLDA уровня логической 1 обеспечит перевод адресной шины в состояние высокого сопротивления. Это в свою очередь достигается переводом тристабильных адресных буферов в третье состояние, т. е. состояние высокого сопротивления.

Кроме того, для обеспечения ПДП в режим высокого сопротивления должна быть переведена и шина данных. В наших системах используется буфер данных, отличающийся от контроллера 8228. Он также должен обладать тремя устойчивыми состояниями, и для его перевода в состояние высокого сопротивления также применяется выходной сигнал микропроцессора 8080 HLDA.

Кроме того, и шина управления посредством сигнала HLDA должна переводиться в состояние высокого сопротивления. Когда система проектируется на основе микропроцессора 8080, нужно точно знать, возникнет ли необходимость в использовании прямого доступа к памяти. Если ПДП использовать предполагается, то все шины системы должны быть снабжены тристабильными буферами. Это позволит передавать функции управления шинами периферийному оборудованию.

6.11. Прямой доступ к памяти в микропроцессоре 8085

Прямой доступ к памяти в микропроцессоре 8085 обеспечивается следующим образом. Вход HOLD микропроцессора 8085, аналогично одноименному входу микропроцессора 8080, должен быть переведен в состояние логической 1. Когда на этом входе установится уровень логической 1, периферийное оборудование может осуществлять прямой доступ к памяти.

Когда микропроцессор 8085 подтверждает получение сигнала HOLD подобно тому, как это делалось в микропроцессоре 8080, выходная линия HLDA микропроцессора 8085 переводится в состояние логической 1. Переход этой линии в состояние логической 1 означает, что микропроцессор 8085 прекратил управление адресной шиной, шиной данных и шиной управления. Указанное действие очень схоже с тем, которое обсуждалось в связи с микропроцессором 8080 при ПДП.

В рассмотренных нами системах адресная шина и шина данных могут иметь буферы, а шина управления может использовать логические TTL-схемы. При этих условиях все три шины системы должны обеспечить возможность перевода их в состояние высокого сопротивления. Сигнал перевода в состояние высокого сопротивления формируется на выходе HLDA микропроцессора 8085. Таким образом, если в проектируемой на базе микропроцессора 8085 системе предполагается использовать ПДП, шины системы должны обладать способностью перехода в состояние высокого сопротивления посредством сигнала с выхода HLDA.

6.12. Прямой доступ к памяти в микропроцессоре Z80

Микропроцессор Z80, подобно микропроцессорам 8080 и 8085, имеет вход, предназначенный для приема сигнала перевода системы в режим ПДП. Для приема запроса на ПДП используется вход $\overline{\text{BUSRQ}}$ микропроцессора Z80. Когда на этой линии

устанавливается уровень логического 0, шина данных, адресная шина и линии шины управления микропроцессора Z80 переходят в состояние высокого сопротивления. После установления этого состояния микропроцессор Z80 выдает сигнал подтверждения (BUSAK) того, что микропроцессор прекратил управлять всеми шинами системы. Эти два сигнала микропроцессора Z80 подобны сигналам HOLD и HLDA микропроцессора 8080.

Укажем еще раз, что для реализации адресных буферов, буферов данных и шины управления могут использоваться стандартные логические TTL-схемы. Если в системе возможен режим ПДП, то логические TTL-схемы должны быть в состоянии переходить в режим высокого сопротивления. Для перевода логических TTL-схем в это состояние в микропроцессоре используется сигнал BUSAK. Действия, выполняемые при этом в системе, подобные тем, которые разбирались выше при рассмотрении ПДП в микропроцессорах 8080 и 8085.

6.13. Прямой доступ к памяти в микропроцессоре 6800

Для обеспечения ПДП в микропроцессоре 6800 используются два основных сигнала. Входной сигнал HALT, вырабатываемый периферийным оборудованием, фактически является запросом на ПДП, поступающим на вход микропроцессора. Когда сигнал HALT достигает уровня логического 0, микропроцессор 6800 прекращает управление адресной шиной, шиной данных и линиями управления. Отказываясь от управления шинами системы, микропроцессор выдает сигнал подтверждения ВА, имеющий уровень логической 1. Тем самым он уведомляет периферийное оборудование о том, что микропроцессор прекратил управлять всеми шинами системы.

При разработке микропроцессорной системы на базе микропроцессора 6800 для реализации интерфейсных схем, адресной шины, шины данных и шины управления могут быть использованы стандартные логические TTL-схемы. В этом случае TTL-схемы должны обеспечить возможность перевода шин системы в состояние высокого сопротивления. Перевод шин в это состояние осуществляется по сигналу ВА, вырабатываемому микропроцессором 6800. Когда сигнал ВА достигает уровня логической 1, адресная шина, шина данных и шина управления переводятся в состояние высокого сопротивления.

При выполнении прямого доступа к памяти в микропроцессорной системе 6800 периферийное оборудование должно подавать на вход HALT микропроцессора сигнал уровня логического 0. Когда линия HALT переходит в состояние логического

0, периферийное оборудование ожидает момента установления состояния логической 1 на выходе ВА микропроцессора 6800. Когда линия выхода ВА устанавливается в состояние логической 1, периферийное оборудование может использовать шины системы, так как они уже находятся в состоянии высокого сопротивления. После осуществления ПДП периферийное оборудование снова переводит линию входа HALT в состояние логической 1. Так выполняется операция ПДП в микропроцессорной системе, построенной на базе микропроцессора 6800.

6.14. Выводы

В настоящей главе мы рассмотрели способы организации прерываний, режима ожидания и прямого доступа к памяти в микропроцессорах 8080, 8085, Z80 и 6800. Кроме того, было продемонстрировано некоторое аппаратное обеспечение, которое может быть использовано для выполнения этих функций. Рассмотренное нами аппаратное обеспечение иллюстрирует частные подходы к реализации конкретных специальных функций. Представленные аппаратные средства не являются единственными или лучшими решениями данной задачи. Можно проектировать многие микропроцессорные системы и не используя информацию, содержащуюся в данной главе. Однако если возникнет необходимость проектировать или восстанавливать микропроцессорную систему, в которой использована хотя бы одна из рассмотренных здесь функций, то материалы настоящей главы окажутся очень полезными. Кроме того, сведения о том, как аппаратные средства участвуют в выполнении этих операций, окажутся полезными при поиске неисправностей в нем. Причем, если мы не знаем общих принципов функционирования аппаратных средств системы во время поиска неисправностей, безусловно будут возникать затруднения. Основательное знание этого материала позволит без особого напряжения проводить анализ каких-либо микропроцессорных систем. Принципы организации прерываний, режима ожидания и ПДП будут сохраняться; реализация же этих функций в разных микропроцессорах может быть различной. Изучив материалы настоящей главы, вы будете хорошо подготовлены для изучения аппаратных средств, реализующих эти функции в других системах.

ПРОГРАММИРОВАНИЕ ПЕРЕПРОГРАММИРУЕМЫХ ПОСТОЯННЫХ ЗАПОМИНАЮЩИХ УСТРОЙСТВ

В гл. 8 будет представлена система, управляемая микропроцессором и осуществляющая программирование перепрограммируемых постоянных запоминающих устройств (ППЗУ). ППЗУ используется во многих микропроцессорных системах. Каждый, кто предполагает серьезно заниматься разработкой микропроцессорных систем и поиском неисправностей в них, должен знать, как программируется и стирается ППЗУ. В настоящей главе мы рассмотрим, каким образом программируется ППЗУ 2708 и как стирается записанная в нем информация.

Обсуждение будем вести на примере ППЗУ 2708, широко распространенном в настоящее время. Изучив принципы программирования ППЗУ 2708, будет не трудно понять, как программируются ППЗУ других типов. Несмотря на то что особенности программирования конкретных ППЗУ будут встречаться всегда, общие принципы останутся неизменными.

7.1. Общие представления о перепрограммируемых постоянных запоминающих устройствах

В вычислительных системах ППЗУ используется точно так же, как и ПЗУ. Основное преимущество ППЗУ заключается в том, что информация в нем может изменяться. В стандартное ПЗУ данные записываются во время его изготовления и потом остаются неизменными. Любое ПЗУ предварительно программируется и будет постоянно содержать фиксированную комбинацию единиц и нулей. ППЗУ не нуждается в предварительном программировании и занесении в него данных. Пользователь сам может занести любую информацию в любые ячейки ППЗУ. Потом, используя специальное оборудование, пользователь может заменить данные, хранимые в ППЗУ.

Возможность изменения данных в ППЗУ является главной причиной применения этого типа устройств памяти. Когда разрабатывается новая микропроцессорная система, первый вариант программного обеспечения целесообразно записать именно в ППЗУ. Затем ППЗУ включается в систему и микропроцессор может выполнять программу, хранимую в ППЗУ. Если программу нужно изменить, то ППЗУ отключаются от системы и

производится желаемое изменение содержащейся в нем информации.

Для программирования ППЗУ и для стирания хранящейся в нем информации необходимо специальное оборудование. Однако это неудобство является незначительным по сравнению с теми преимуществами, которые предоставляет ППЗУ пользователям микропроцессорных систем.

7.2. Стирание информации в перепрограммируемых постоянных запоминающих устройствах

Указывалось, что информация, содержащаяся в ППЗУ, может быть изменена. Один способ изменения информации в ППЗУ заключается в стирании всех данных, записанных в него ранее. Стирание информации в ППЗУ 2708 состоит просто в установке всех разрядов ППЗУ в состояние логической 1. Это значит, что после выполнения этой операции чтение данных из ППЗУ обнаруживает во всех его ячейках только логические 1.

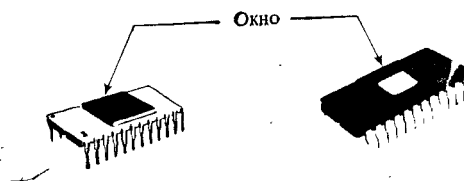


Рис. 7.1. Два различных способа реализации ППЗУ. «Окно» в каждом модуле пропускает ультрафиолетовое излучение, которое используется для стирания данных в ППЗУ. (С разрешения фирмы Intel.)

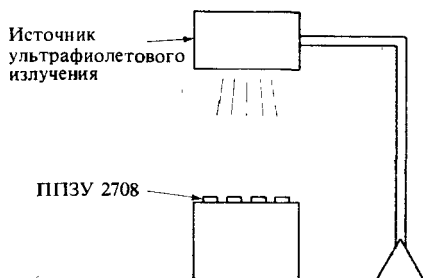
Существуют различные схемы для выполнения стирания данных в ППЗУ. Одна из них — источник ультрафиолетового излучения. Конструкция ППЗУ 2708 в верхней его части предусматривает «окно», через которое ультрафиолетовое излучение может проникать в устройство (рис. 7.1). Чтобы осуществить затирание информации в устройстве 2708, следует поместить его на определенный период времени рядом с источником ультрафиолетового излучения. Примерное время очистки ППЗУ составляет около получаса, однако оно зависит от многих факторов.

Источником ультрафиолетового излучения служит специальное устройство. Оно необходимо каждому, кто предполагает активно использовать ППЗУ. На рис. 7.2 изображена схема типичной установки для выполнения операции стирания данных в ППЗУ.

Как отмечалось, после выполнения стирания информации все ячейки ППЗУ находятся в состоянии логической 1. Зная это, всегда можно осуществить проверку правильности прове-

дения операции очистки. Для этого достаточно прочитать содержимое каждой ячейки ППЗУ и убедиться в том, что оно равно логической 1. Этот частный вопрос будет обсуждаться в гл. 9, там же будут рассмотрены программные средства, предназначенные для управления аппаратными средствами, представленными в гл. 8. К сказанному добавим, что некоторые ППЗУ после стирания в них информации содержат во всех

Рис. 7.2. Схема типичной установки для стирания данных в ППЗУ.



ячейках логические нули. Поэтому всегда следует ознакомиться с техническими характеристиками используемого ППЗУ.

Оставшаяся часть главы посвящается некоторым аспектам вопроса программирования ППЗУ, т. е. обсуждению того, как данные записываются в ППЗУ. Запись данных в ППЗУ некоторым образом изменяет внутренние характеристики устройства памяти. Именно изменения характеристик приводят к тому, что в выбранных ячейках памяти запоминаются значения, равные логическому 0. Итак, в память могут записываться только значения, равные логическому 0; запись же логических значений, равных 1, производится лишь при стирании содержимого памяти с помощью ультрафиолетового излучения.

Процесс изменения внутренних характеристик устройства 2708 называют *переносом заряда*. При записи информации в устройство 2708 происходит определенное накопление зарядов в кристалле кремния. Таким образом, путем накопления зарядов в определенной ячейке памяти логическое значение 1 заменяется на логическое значение 0. Когда осуществляется стирание информации в ППЗУ, скопления зарядов, образованные во время записи информации, рассеиваются.

Теперь вы имеете очень поверхностное, но вполне достаточное для пользователей представление о процессах, происходящих на атомном уровне в кристалле кремния.

7.3. Программирование ППЗУ 2708

Теперь, ссылаясь на временную диаграмму, рассмотрим последовательность действий, выполняемых в процессе программирования устройства 2708. Программирование ППЗУ 2708 за-

ключается в записи в него данных по указываемым адресам. Записываемые данные подаются на выходы устройства, а адрес, по которому производится запись, устанавливается на адресных входах. Затем на определенные выходы устройства подаются необходимые уровни напряжения. Последовательность

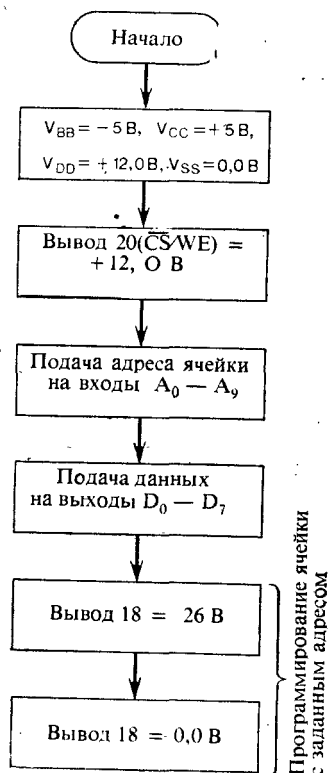


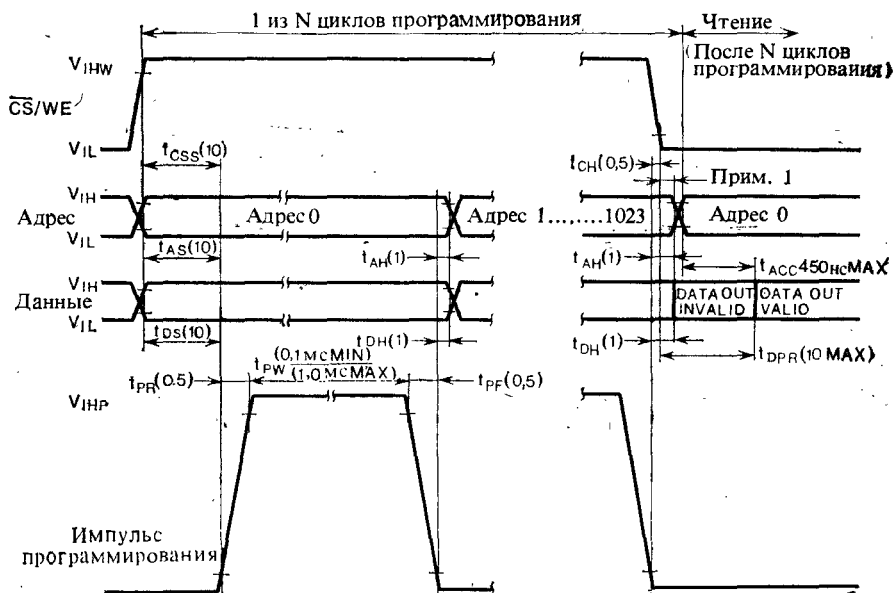
Рис. 7.3. Блок-схема последовательности действий при записи информации в одну ячейку ППЗУ.

действий, выполняемых при записи данных в ППЗУ, приведена на рис. 7.3 в виде блок-схемы.

Временная диаграмма, представленная на рис. 7.4, демонстрирует временные соотношения между сигналами, отображающими данные и адреса, и импульсами программирования, подаваемыми на устройство 2708. Как указано в технических характеристиках устройства 2708, его адресные входы и входы данных рассчитаны на стандартный уровень напряжения, установленный для TTL-схем. Сигнал $\overline{CS/WE}$, подаваемый на вывод 20 устройства 2708, в течение времени программирования устройства 2708 должен иметь уровень $+12 \text{ В}$. Импульс программирования, подаваемый на вывод 18 ППЗУ 2708, изменяет

Характеристики сигналов, используемых при программировании ППЗУ

Обозначение	Параметр	Значение			Единица измерения
		мин.	тип.	макс.	
t_{AS}	Время установки адреса	10			мкс
t_{CSS}	Время установки сигнала CS/WE	10			мкс
t_{DS}	Время установки данных	10			мкс
t_{AH}	Время фиксации адреса	1			мкс
t_{CH}	Время фиксации сигнала CS/WE	0,5			мкс
t_{DH}	Время фиксации данных	1			мкс
t_{DPR}	Время задержки чтения данных из ППЗУ по отношению к моменту окончания цикла программирования			10	мкс
t_{PW}	Ширина импульса программирования	0,1		1,0	мс
t_{PR}	Время переднего фронта импульса программирования	0,5		2,0	мкс
t_{PF}	Время заднего фронта импульса программирования	0,5		2,0	мкс



Примечание 1: задний фронт сигнала $\overline{CS/WE}$ должен проходить после прохождения заднего фронта программного импульса и перед подачей кода адреса.

Примечание 2: числа, помещенные в круглые скобки, если особо не оговаривается, указывают временной интервал, выраженный в мкс.

Рис. 7.4. Форма сигналов, используемых при записи данных в ППЗУ. (С разрешения фирмы Intel.)

ся от уровня 0 до уровня 26 В. Он нарастает до 26 В, а затем снижается до нулевого уровня для каждой очередной программируемой ячейки ППЗУ. При программировании устройства 2708 для записи данных в ячейку с определенным адресом требуется подать один импульс программирования. Таким обра-

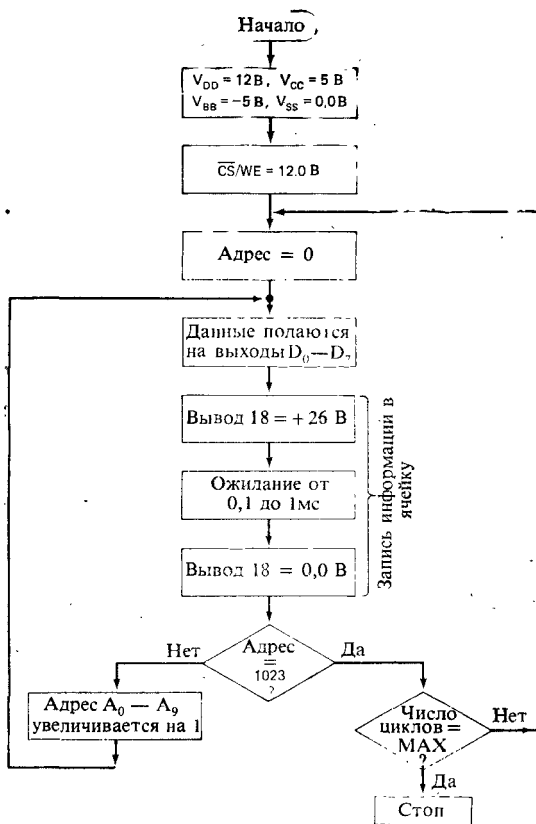


Рис. 7.5. Блок-схема последовательности действий, выполняемых при программировании ППЗУ.

зом, при программировании устройства 2708 потребуется выполнить 1024 цикла, т. е. столько, сколько имеется в ППЗУ ячеек.

На рис. 7.5 изображена блок-схема, определяющая последовательность действий, выполняемых в одном цикле программирования ППЗУ 2708.

Анализируя блок-схему, видим, что уровень сигнала $\overline{\text{CS}}/\text{WRITE ENABLE}$ устанавливается равным +12 В до того,

как данные будут поданы на выходы устройства. С помощью этого сигнала выходы устройства 2708 переводятся в режим ввода данных. Это действие позволяет предотвратить конфликт данных, имеющихся на выходах устройства, с записываемыми в него данными. Кроме того, все разряды вводимых данных подаются на устройство 2708 параллельно. Одновременно производится программирование восьми разрядов ППЗУ.

7.4. Импульс программирования ППЗУ 2708

Импульс программирования, подаваемый на устройство 2708, при уровне в $+26$ В имеет минимальную длительность $0,1$ мс и максимальную длительность, равную 1 мс. Однако фактическое значение ширины этого импульса может быть любым в заданных пределах. Хотя ширина импульса и может принимать в указанных пределах произвольное значение, важно, как будет показано ниже, знать точное значение ширины импульса программирования.

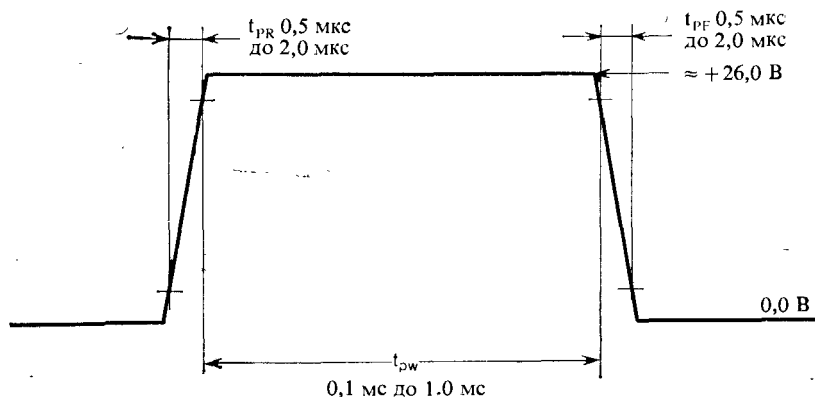


Рис. 7.6. Форма импульса программирования для ППЗУ 2708.

Другими важными характеристиками импульса программирования являются длительности переднего и заднего фронта. Эти характеристики должны находиться в пределах от $0,5$ до $2,0$ мкс. На рис. 7.6 показана форма импульса и указаны его характеристики. Для программирования устройства 2708, согласно спецификациям, необходимо обеспечить выполнение следующего условия: произведение ширины импульса программирования на число циклов, выполняемых при программировании памяти, должно быть не менее 100 мс. В любом цикле программирования при переходе к каждой следующей ячейке ППЗУ подается импульс программирования. Таким образом, необходимое количество циклов программирования памяти

можно определить следующей формулой:

$$NP = \frac{100 \text{ мс}}{\text{Длительность импульса программирования в мс}} \cdot$$

В нашей системе длительность импульса программирования равна 0,5 мс. Следовательно, потребуется выполнить следующее число циклов:

$$NP = \frac{100 \text{ мс}}{0,5 \text{ мс}} = 200,$$

Если выполнять программирование устройства согласно спецификациям, то нужно сделать 200 циклов, в каждом из которых обрабатываются все ячейки ППЗУ. Можно выполнить и больше чем 200 циклов. При этом только возрастет время программирования. Увеличение количества циклов не улучшает качества программирования, и поэтому целесообразно выполнять количество циклов, определяемое приведенной выше формулой. Однако многие пользователи считают, что, чем больше время программирования ППЗУ, тем выше будет качество программирования. Это ошибочная точка зрения, и поэтому следует придерживаться рекомендованных правил.

В процессе программирования памяти может обнаружиться, что она уже содержит информацию, которую и предполагалось в нее записать, а предписанное число циклов программирования еще не исчерпано. Например, мы могли убедиться в том, что уже после 20 циклов вместо 200, рекомендуемых инструкцией, в памяти содержатся желаемые комбинации 0 и 1. Однако в этом случае во избежание последующего самопроизвольного разрушения записанной информации следует продолжать программирование в течение заданного времени.

Вначале мы упоминали, что изменение внутренних характеристик ППЗУ 2708 основано на эффекте накопления заряда. Если уже накоплено некоторое минимальное количество заряда, достаточное для изменения внутренних характеристик ППЗУ (затраченное при этом на программирование время окажется меньше номинального), то рассеивание некоторых зарядов может изменить состояние памяти, т. е. записанную туда информацию. Таким образом, если уменьшить время программирования ППЗУ, то во время его использования может произойти разрушение хранимой в памяти информации, т. е. вместо записанного логического значения 0 может появиться логическое значение 1. Следовательно, программировать ППЗУ нужно всегда в течение предписанного времени.

Когда накопится заряд, соответствующий номинальному времени программирования, рассеивание некоторых зарядов не будет приводить к искажению информации в ППЗУ, так как остающееся число зарядов будет достаточным для сохранения

определенного состояния памяти. Не стоит жертвовать надежностью хранения информации в ППЗУ ради ускорения процесса программирования.

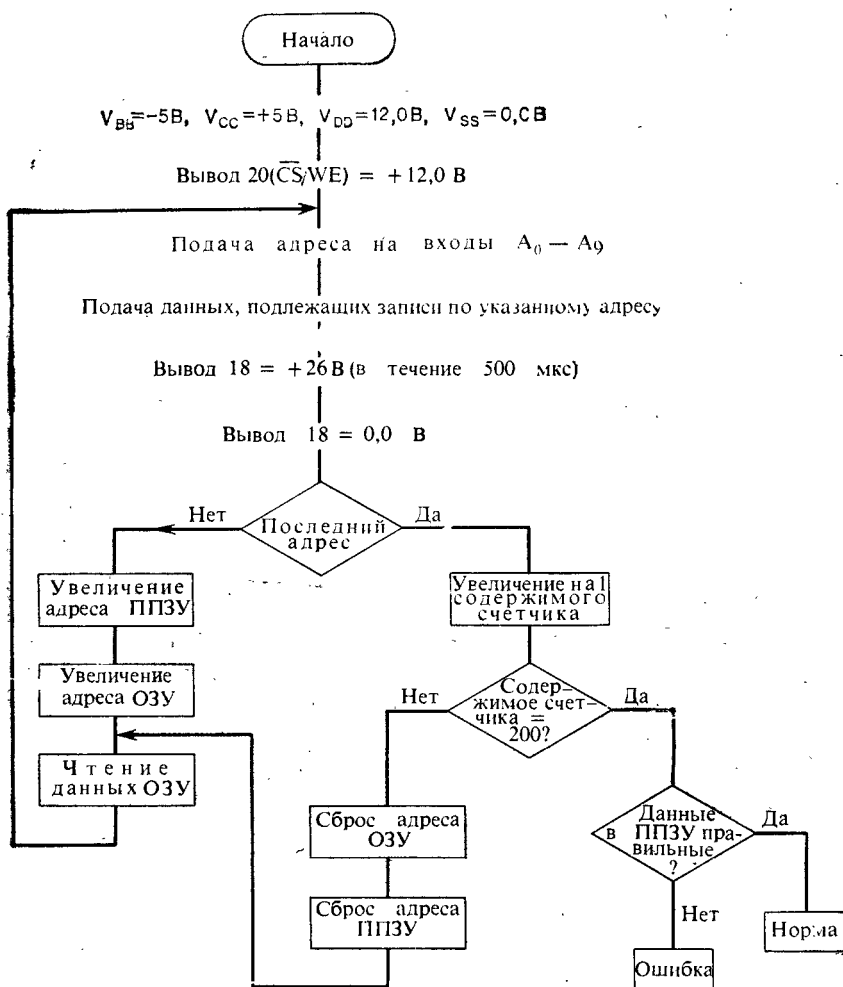


Рис. 7.7. Блок-схема процесса программирования ППЗУ 2708 с использованием ОЗУ. Предполагается, что данные были предварительно записаны в ОЗУ.

В гл. 8 будет рассмотрена система, которая реализуется в соответствии с блок-схемой, представленной на рис. 7.7. Мы видим, что данные, предназначенные для записи в ППЗУ, предварительно помещаются во внешнее ОЗУ, а уже из ОЗУ пе-

редаются на входы ППЗУ 2708. Сравнение содержимого ОЗУ с данными, читаемыми из ППЗУ, производится, когда требуемое число циклов программирования завершится. Если при выполнении сравнения содержимого всех соответствующих ячеек ОЗУ и ППЗУ установлено полное совпадение, то, значит, ППЗУ запрограммировано успешно.

7.5. Выводы

В настоящей главе рассмотрены некоторые важные характеристики ППЗУ 2708 и принципы программирования этого типа устройства памяти. Материалы, представленные здесь, будут использованы в следующей главе при проектировании микропроцессорной системы, предназначенной для автоматического программирования ППЗУ 2708. Зная, как программируется ППЗУ 2708, вы сможете без большого труда разобраться в особенностях программирования других подобных устройств.

Глава 8

ТЕХНИЧЕСКИЕ СРЕДСТВА УСТРОЙСТВА ПРОГРАММИРОВАНИЯ ППЗУ

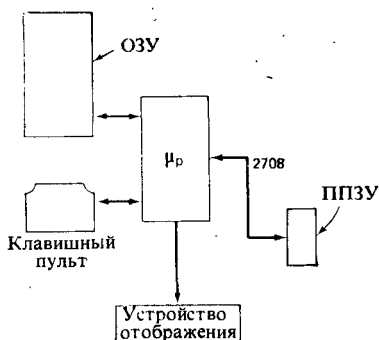
В настоящей главе рассматриваются технические средства микропроцессорной системы, используемой для автоматического программирования ППЗУ 2708. Вопросы программирования устройства 2708 рассматривались ранее в гл. 7, однако там не затрагивалась автоматизация этого процесса. Для построения описываемых систем будут использованы четыре типа микропроцессоров. Вначале рассматриваются системы на базе микропроцессоров 8080, 8085 и Z80; затем обсуждение связано с системами на основе устройства 6800.

В настоящей главе описывается лишь проектирование технических средств системы. Соответствующие программные средства будут рассмотрены в гл. 9. В гл. 10 обсуждаются вопросы отладки системы с помощью техники тестирования посредством статических сигналов.

8.1. **Общее описание системы**

Прежде чем обсуждать особенности построения системы, необходимо определить ее предназначение. Основная функция рассматриваемой системы заключается в программировании ППЗУ

Рис. 8.1. Структурная схема технических средств системы автоматизации программирования ППЗУ.



2708. Однако такое широкое определение функции системы порождает ряд вопросов.

Первый из них связан с определением устройства для хранения данных для программирования ППЗУ. В качестве подобного устройства может быть использовано ОЗУ. Однако при этом возникает другой вопрос: каким образом эти данные вводятся в ОЗУ? Естественным решением является ввод данных с помощью клавишного пульта.

При этом возникает проблема визуального контроля вводимых данных, которая может быть решена путем использования 6-значного шестнадцатеричного табло (устройства отображения).

Теперь, когда уже определены некоторые общие блоки предлагаемой системы, необходимо выбрать способы их реализации. Эти блоки показаны на рис. 8.1, который отображает общую функцию системы в виде этапов: а) ввода данных в ОЗУ посредством клавишного пульта и б) программирования устройства 2708 с помощью введенных данных.

8.2. Специфические функции системы

Для ввода данных посредством клавишного пульта необходимо реализовать следующий набор ключей:

1. Ключи данных 0—F.
2. Ключ ввода.
3. Ключ стирания ввода.
4. Ключ установки адреса.
5. Ключ стирания ОЗУ.
6. Ключ программирования.
7. Ключ верификации (VFY).
8. Ключ копирования.
9. Ключ сброса системы.

Ниже определено назначение этих ключей.

КЛЮЧИ ДАННЫХ: шестнадцать ключей, используемых для ввода данных в систему и обозначаемых как 0—F.

КЛЮЧ ВВОДА: используется для «уточнения» вводимых данных путем их сверки.

КЛЮЧ СТИРАНИЯ ВВОДА: используется для стирания данных, введенных с помощью ключей данных, либо для стирания установленного адреса.

КЛЮЧ СТИРАНИЯ ОЗУ: подготавливает к работе внутреннее ОЗУ путем установки единичных значений данных.

КЛЮЧ ПРОГРАММИРОВАНИЯ: переводит систему в режим программирования, при котором данные из ОЗУ передаются в устройство 2708, с высвечиванием соответствующей индикации режима на табло. После нажатия этого ключа останов работы системы возможен лишь путем нажатия на ключ сброса.

КЛЮЧ ВЕРИФИКАЦИИ (VFY): при нажатии этого ключа производится сравнение данных в устройстве 2708, размещенном на панели, с данными, хранимыми в ОЗУ. При обнаружении несовпадения на табло высвечивается адрес и значения данных в ППЗУ 2708.

КЛЮЧ КОПИРОВАНИЯ: при нажатии этого ключа осуществляется дублирование данных из ППЗУ 2708 на панели во внутреннее ОЗУ. Этот режим используется для получения идентичных ППЗУ (на основе уже имеющихся).

КЛЮЧ СБРОСА: при нажатии ключа прекращается выполнение очередной команды и начинается выполнение программы инициализации.

При включении система выполняет ту же последовательность команд, что и при нажатии ключа сброса.

8.3. Вспомогательные технические средства

Теперь, когда стало ясно предназначение системы, можно приступить к проектированию аппаратуры, реализующей рассмотренные функции.

Для штатного устройства 2708 в режиме чтения, когда последнее функционирует подобно ПЗУ, необходимы напряжения 12, 5 и -5 В. При работе в режиме программирования необходимо дополнительно обеспечить импульс программирования напряжением 26 В. Поэтому для работы устройства требуются уровни напряжения в 26, 12, 5 и -5 В. Используемые в системе TTL-схемы также требуют напряжения 5 В при номинальном токе, величину которого нетрудно вычислить. Существует большое число источников питания с напряжением 5 В, которые можно использовать в системе. Остановимся на источнике, обеспечивающем ток 3 А при напряжении 5 В.

Для напряжения -5 В характерна малая величина тока, обычно порядка 200 мА. Из технических характеристик устройства 2708 видно, что наибольший ток для V_{BB} составляет 60 мА. Предполагается, что в системе будут использованы три таких устройства: два для управления системой и одно для программирования. Кроме того, необходимо предусмотреть возможность расширения системы. Поэтому расчеты следует проводить для четырех ППЗУ 2708. При этом наибольший ток для V_{BB} будет составлять $4 \times 60 = 240$ мА. Следовательно, выбирается стандартный источник, обеспечивающий при -5 В ток силой 500 мА.

От напряжения 12 В должны питаться два устройства 2708, управляющие системой, и программируемое устройство. Если рассчитывать на расширение системы до 4 устройств, используемых для управления, то с учетом программируемого устройства источник в 12 В должен обеспечить ток $5 \times 65 = 325$ мА. Здесь значение 65 мА представляет «наихудший случай» для тока I_{DD} в соответствии с техническими характеристиками устройства 2708.

Поскольку для программирования необходимо 26 В, то напряжение 12 В можно обеспечить с помощью регулятора от источника в 26 В. Наибольшая величина рассеиваемой мощности для такого регулятора определяется как $(26 \text{ В} - 12 \text{ В}) \times I_{DD}$ (системы) и составляет $P_D = 14 \text{ В} \times 325 \text{ мА} = 4,55 \text{ Вт}$.

В качестве регулятора используем устройства LM340C-12. Получение требуемых напряжений показано на рис. 8.2.

При использовании микропроцессора 8080 значения тока при напряжении 12 В и —5 В повышаются, поскольку тактовый генератор 8224 также потребляет 12 В.

Промышленность выпускает целый ряд источников питания с различными значениями напряжения и тока. После оценки потребностей системы обычно можно подобрать один из таких источников, обеспечивающий требуемые спецификации. Если же проектировщик намерен создать свой собственный источник

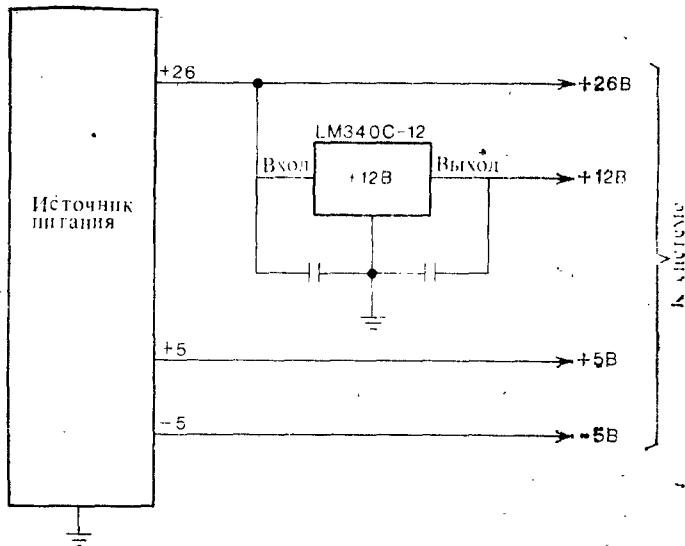


Рис. 8.2. Схема подачи питания в системе.

питания, то при разработке его можно использовать регуляторы напряжения с тремя зажимами. Однако предпочтительнее все же применять готовый источник.

8.4. Клавишное устройство ввода-вывода

Для построения клавишного устройства используем технические средства, описанные в гл. 4. Схема устройства, рассчитанная на 24 ключа, представлена на рис. 8.3. Реализованы следующие ключи:

16 ключей данных (для набора цифр от 0 до F₁₆)

- 1 ключ ввода
- 1 ключ стирания ввода
- 1 ключ установки адреса
- 1 ключ стирания ОЗУ
- 1 ключ программирования
- 1 ключ верификации
- 1 ключ копирования
- 1 ключ сброса системы

Функциональные ключи

Итого 24 ключа

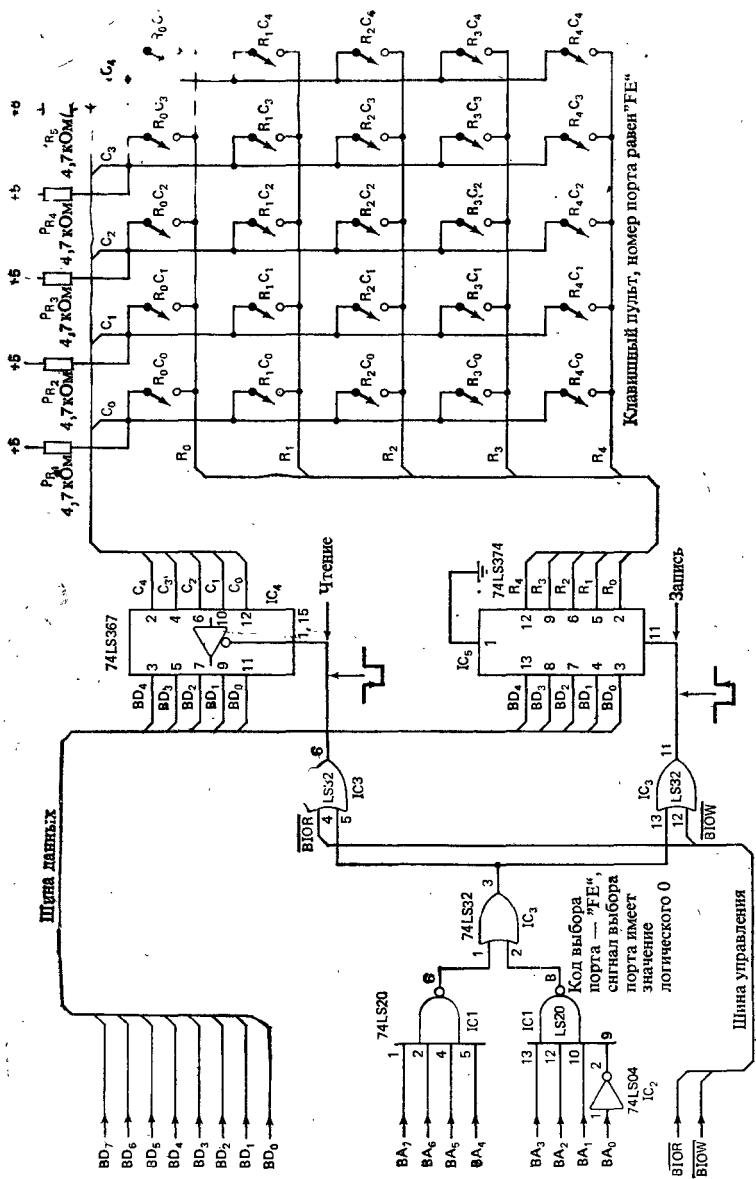


рис. 8.3. Полная схема интерфейса клавишного пульта системы. Пунктирной линией выделены физические переключатели пульта.

Ключ сброса системы не входит в наборную матрицу. Функция сброса не синхронизирована с другими функциями и реализована с помощью отдельного переключателя, тип и расположение которого описаны ниже при рассмотрении технических средств системы в целом.

8.5. Технические средства устройства отображения

Будем использовать то же самое устройство отображения, которое было рассмотрено в гл. 4. Принципиальная схема этого устройства приведена на рис. 8.4.

8.6. Постоянная память системы

Постоянную память системы можно построить путем использования ППЗУ 2708 со стиранием информации. Для реализации всех команд системы применяются два устройства 2708; однако при проектировании системы для обеспечения возможности ее расширения предусматривается наличие четырех ППЗУ 2708. В ПЗУ системы принято следующее распределение памяти:

Адресное пространство

0000—03FF	ПЗУ 0	}	В этих ПЗУ хранятся программы
0400—07FF	ПЗУ 1		
0800—0BFF	ПЗУ 2	}	Данные ПЗУ не используются, а предназначены для расширения системы
0C00—0FFF	ПЗУ 3		

8.7. Оперативная память системы

Хранение данных для программирования устройства 2708 осуществляется в ОЗУ объемом 1 К (1024 байт). Для работы микропроцессора необходима дополнительная память ОЗУ, чтобы разместить «стек» и временно запомнить данные. Используемое в системе ОЗУ состоит из четырех устройств статической памяти 2114 объемом 1 К. Два из этих устройств включают ОЗУ 1 К×8. Ниже приведено распределение памяти ОЗУ системы:

Адресное пространство

1000—13FF	Память системы, используемая микропроцессором
1400—17FF	Память системы, предназначенная для программирования 2708
1800—1BFF	Не используется, память предназначена для возможного расширения системы

Отметим, что как для ПЗУ, так и для ОЗУ дополнительный объем закладывается при проектировании. Это позволяет лег-

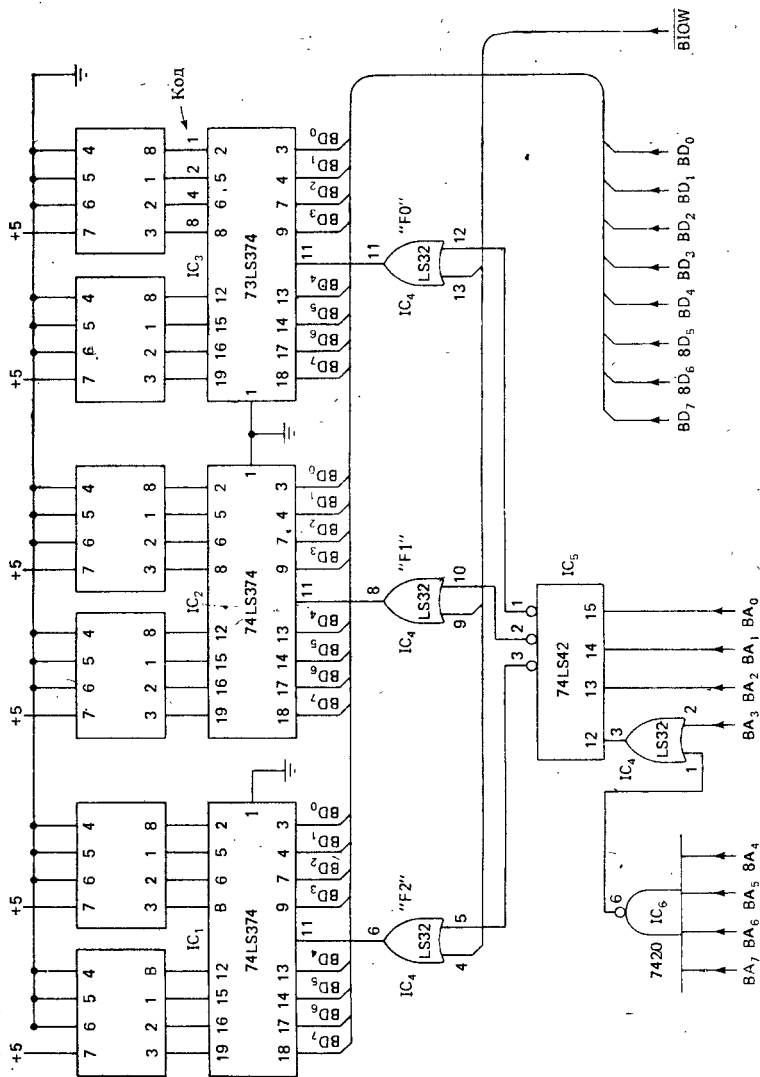


Рис. 8.4. Полная схема устройства отображения системы;

ко приспособиться к изменениям, которые могут возникнуть в будущем. Отметим также, что распределение памяти, показанное для ПЗУ и ОЗУ системы, используется лишь в случае применения микропроцессоров 8080, 8085 и Z80. Для микропроцессора 6800 требуется другое распределение памяти вследствие особенностей интерпретации функции сброса. Реализация системы, использующей устройство 6800, обсуждается ниже.

8.8. Интерфейс программирования 2708

Рассмотрим, какие средства необходимы для обеспечения поступления на вход требуемой информации для программирования устройства 2708. Это устройство имеет 10 адресных линий.

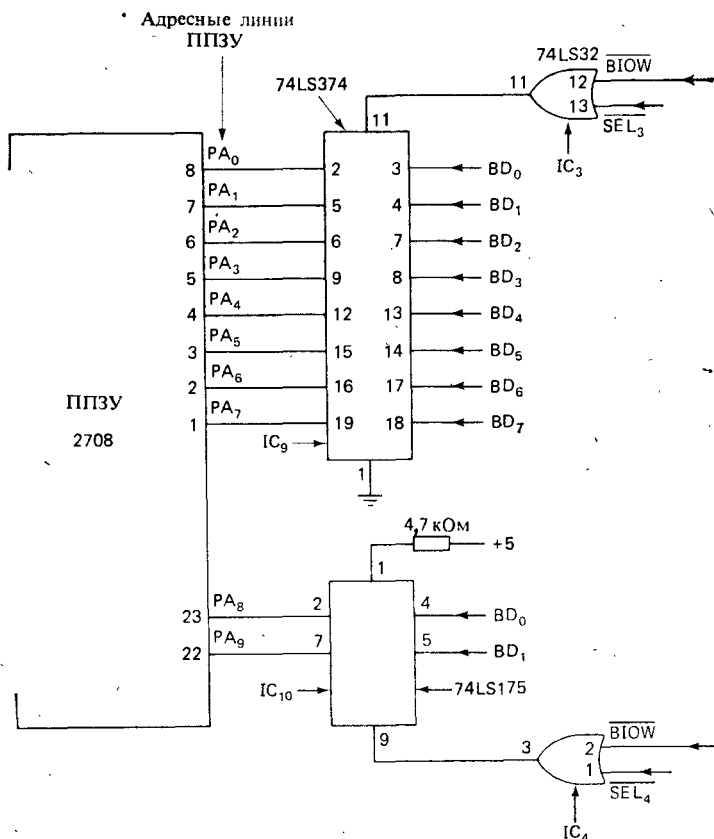


Рис. 8.5. Схема выдачи адреса на программируемое ПЗУ.

Представим подачу адреса на устройство 2708 с помощью двух отдельных выходных портов. Один из них предназначен для адресных входов A₀—A₇, другой — для входов A₈ и A₉. Для ре-

ализации этих портов используем 8-разрядный фиксатор 74LS374 и 4-разрядный фиксатор 74LS175. Адреса этих устройств будут соответственно 00_{16} и 01_{16} , где 00_{16} относится к адресным линиям $A_0—A_7$, 01_{16} — к линиям $A_8—A_9$ устройства 2708.

Технические средства, необходимые для реализации описанной функции адресования, приведены на рис. 8.5.

8.9. Технические средства ввода и вывода данных

Для программирования ППЗУ 2708 необходимо подать информацию на выводы данных. Для проверки правильности программирования устройства 2708 данные считываются из ППЗУ.

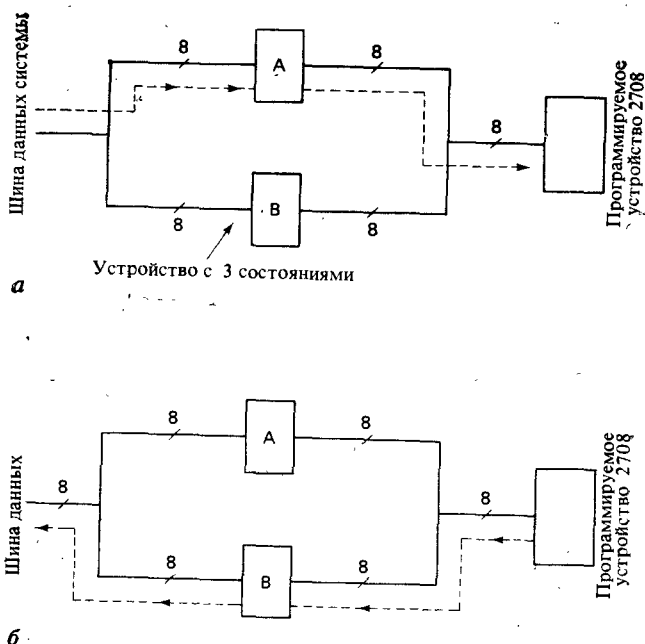


Рис. 8.6. Схема передачи данных из системы в программируемое ППЗУ (а); схема передачи данных от программируемого ППЗУ в систему (б).

Таким образом, в отдельные промежутки времени данные считываются, в другие — записываются в ППЗУ 2708. Для ввода данных в устройство 2708 используется системный адрес вывода 02_{16} , для вывода — системный адрес ввода 02_{16} .

При подаче данных в ППЗУ и при чтении данных из ППЗУ необходимо отделить их от данных на выходах устройства. Это показано на блок-схемах рис. 8.6, а и б.

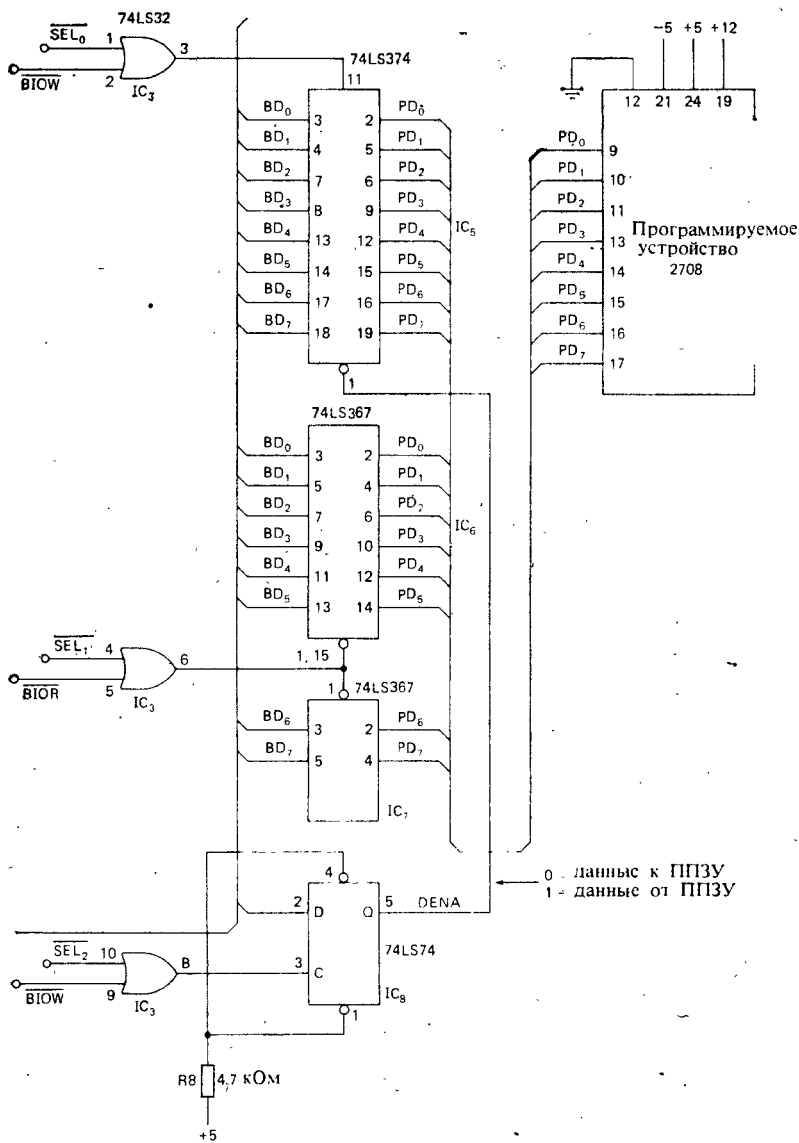


Рис. 8.7. Полная схема реализации двунаправленной передачи данных для программируемого ППЗУ. Принцип работы схемы показан на рис. 8.6.

Чтобы не допустить взаимосвязь технических средств блоков А и В на рис. 8.6, а и б, необходима линия управления, определяющая состояния чтения и записи данных для устройства 2708. Для микропроцессорной системы эта линия будет играть роль устройства вывода. Код выбора вывода для этого устройства есть 03₁₆. На рис. 8.7 показана схемная реализация описанной функции управления.

8.10. Средства формирования импульса программирования

Уровень напряжения для импульса программирования на выводе 18 ППЗУ 2708 меняется от +26 В до 0 В. Продолжительность импульса составляет примерно 0,5 мс. Сигнал управления изменением уровня импульса программирования (от высокого +26 В до низкого 0 В) должен соответствовать уровню TTL-схем. Это отражено на рис. 8.8, а и б.

Из рис. 8.8, б следует необходимость построения схемы сдвига уровня напряжения от уровня TTL до +26 В. Микропроцессор выдает соответствующим образом синхронизированный TTL-импульс для управления импульсом программирования. На рис. 8.9 приведена схема генерации импульса высокого уровня, в которой в качестве разрешающего сигнала используется TTL-сигнал управления. Сигнал ENABLE является дополняющим для разрешающего TTL-сигнала. Ниже будет рассмотрена генерация сигналов разрешения и ENABLE. Если сигнал разрешения на рис. 8.9 соответствует логической 1, то сигнал на выводе 2 инвертора со свободным коллектором 7406 соответствует логическому 0. В результате в цепи источника 26 В через резисторы R_1 и R_2 на землю проходит ток. В этих условиях открывается транзистор T_1 , так что потенциал коллектора стремится к +26 В. Уровень сигнала ENABLE соответствует логическому 0. Вывод 4 устройства 7406 выступает как свободный коллектор. В результате чего потенциал точки 4 повышается до +26 В. Временем нарастания импульса управляет резистор R_3 и конденсатор C_1 .

Положим, что разрешающий сигнал рис. 8.9 соответствует логическому 0. Это будет означать, что сигнал ENABLE соответствует логической 1. Вывод 2 инвертора 7406 будет открыт вследствие подачи сигнала логического 0 от линии разрешения. Поэтому транзистор T_1 закрывается. Сигнал на выводе 4 инвертора 7406 соответствует логическому 0 при сигнале 1 на входе ENABLE. В результате потенциал вывода А соответствует 0 В. При этом время спада выходного импульса управляется резистором R_4 и конденсатором C_1 . Из изложенного ясно, каким образом схемы TTL управляют генерацией импульса программирования +26 В.

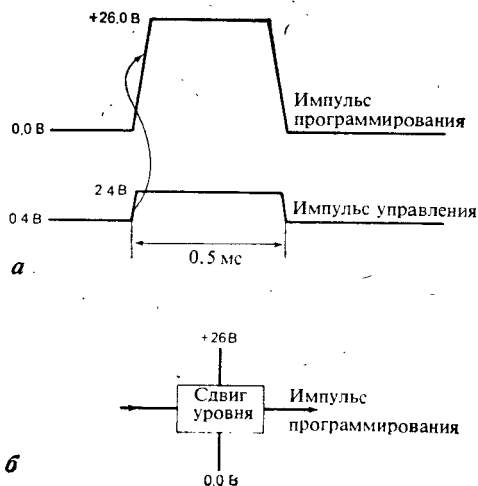


Рис. 8.8. Временная диаграмма формирования импульса программирования высокого уровня напряжения в сравнении с управляющим TTL-импульсом (а); схема сдвига уровня, осуществляющая преобразование TTL-импульса управления в импульс программирования высокого уровня, необходимый для программирования ППЗУ (б).

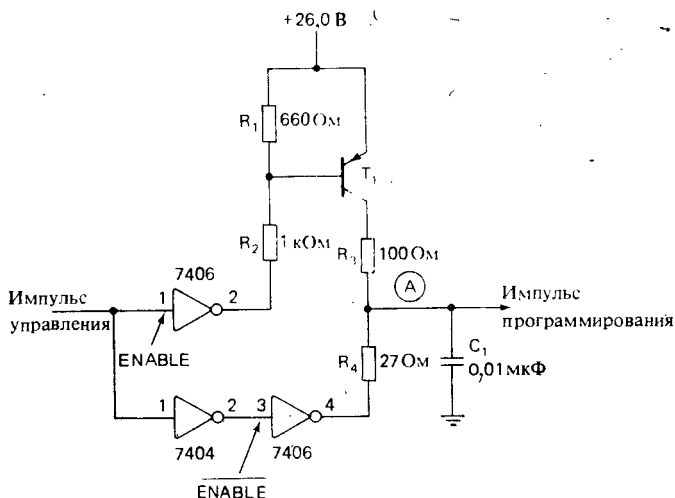


Рис. 8.9. Полная схема сдвига уровня, построенная на основании блок-схемы рис. 8.8, б.

8.11. Средства управления уровнем напряжения на выводе $\overline{CS}/WE$

При программировании ППЗУ 2708 вывод 20 $\overline{CS}/WE$ должен быть соединен с источником $+12$ В. При этом уровень напряжения также должен быть TTL-управляемым. Используем тот же подход, что применялся при генерации импульса программирования. Соответствующая схема приведена на рис. 8.10. Отметим, что потенциал эмиттера на схеме для транзистора T_2 соответствует $+12$ В. При этом отсутствует конденсатор для контроля времени нарастания и спада сигнала.

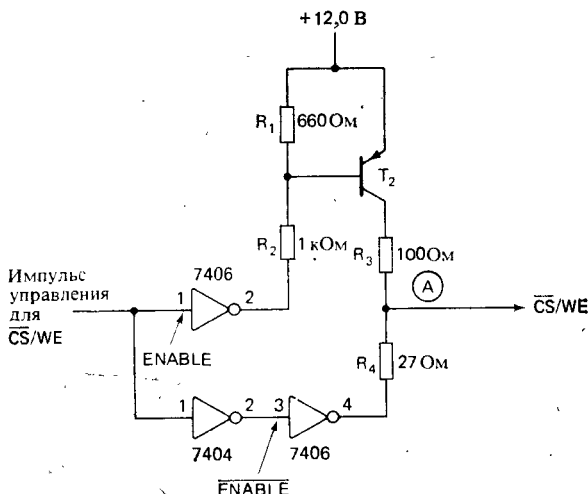


Рис. 8.10. Полная схема генерации сигнала $\overline{CS}/WE+12$ В для программируемого ППЗУ.

Таким образом, было показано, как генерируются сигналы и импульс программирования. Рассмотрим теперь, как микропроцессорная система управляет разрешающими сигналами этих схем. С этой целью указанные схемы рассматриваются как устройство вывода, имеющее адрес 04_{16} . Управление импульсом программирования осуществляется посредством разряда D_0 , а управление импульсом $\overline{CS}/WE$ — посредством разряда D_1 . Схема подобного устройства вывода приведена на рис. 8.11.

8.12. Источник питания для ППЗУ 2708

Для программирования ППЗУ 2708 необходимо обеспечить подачу постоянного напряжения $+12$ В на вывод 19, $+5$ В на вывод 24, -5 В на вывод 21 и 0 В (заземление) на вывод 12. Соответствующие уровни напряжения должны быть подведены к панели программирования. К каждому из указанных вы-

8.14. Использование микропроцессора 6800 в качестве управляющего устройства

При использовании микропроцессора 6800 требуется иное, чем в случае микропроцессоров 8080, 8085 или Z80, распределение памяти. Такое распределение памяти приведено на рис. 8.13. Его особенность заключается в том, что при сбросе системы используются данные с адресами FFFE и FFFF. Поэтому для ПЗУ отводится верхняя часть адресного пространства.

FC00 — FFFF	ПЗУ ₁	} Программы системы
F800 — FBFF	ПЗУ ₀	
F400 — F7FF	ПЗУ ₂	
F000 — F3FF	ПЗУ ₃	
EC00 — EFFF	ОЗУ ₁	Не используется
E800 — EBFF	ОЗУ ₂	Системное ОЗУ
		ОЗУ данных для программирования
E400 — E7FF	ОЗУ ₃	Не используется
E000 — E3FF	Системный ввод-вывод	

Рис. 8.13. Распределение памяти для системы, использующей микропроцессор 6800 в качестве контроллера.

Адресация ввода-вывода построена по аналогии с обращением к памяти. Операция ввода-вывода выполняется при выборе адреса, находящегося в промежутке E000 и E3FF. Все коды выбора ввода-вывода можно оставить такими, какими они были приняты для системы 8085. Для устройства отображения резервируются адреса E0F0, E0F1, E0F2 вместо F0, F1, F2, принятых для системы 8080.

На рис. 8.14 показана генерация сигналов выбора памяти. Здесь же приведен сигнал $\overline{IO}/M$, также используемый для формирования шины управления. Схема шины управления представлена на рис. 8.15. Отметим, что генерирование используемого здесь сигнала $\overline{IO}/M$ показано на рис. 8.14. Кроме того, отметим, что уточнение некоторых сигналов линии управления осуществляется посредством тактового генератора фазы 2. Полная схема системы 6800 представлена на рис. 8.16, 8.17 и 8.18.

8.15. Выводы по техническим средствам

В данной главе были описаны технические средства, необходимые для построения системы, управляемой микропроцессором. Рассматриваемая система предназначалась для программирования ППЗУ. Однако при обсуждении особенностей проектирования этой системы были рассмотрены стандартные приемы построения интерфейса, применимые к любым системам, управляемым микропроцессором.

Был представлен и обсужден полный состав схем, необходимых для построения устройства программирования ППЗУ с

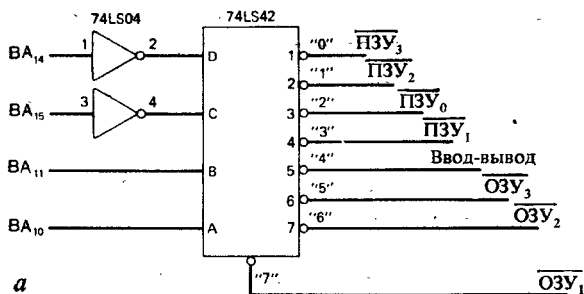
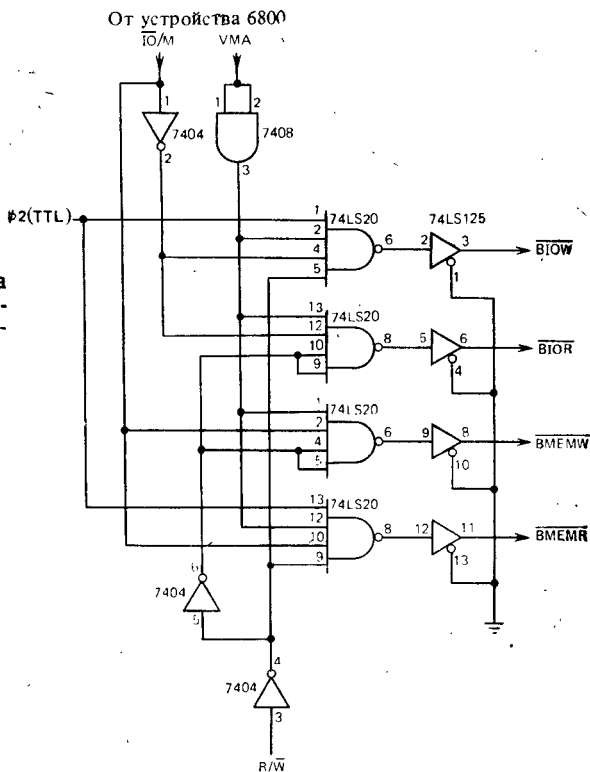


Рис. 8.14. Схема генерации кодов выбора устройства для системы 6800. Необходимо обратить внимание на генерацию сигнала разрешения ввода-вывода.

				Функция	Состояние устройства 74LS42
A ₁₅	A ₁₄	A ₁₁	A ₁₀		
1	1	0	0	ПЗУ ₃	0
1	1	0	1	ПЗУ ₂	1
1	1	1	0	ПЗУ ₀	2
1	1	1	1	ПЗУ ₁	3
0	1	0	0	В/В	4
0	1	0	1	ОЗУ ₃	5
0	1	1	0	ОЗУ ₂	6
0	1	1	1	ОЗУ ₁	7

б

Рис. 8.15. Полная схема реализации сигналов шины управления системы 6800.



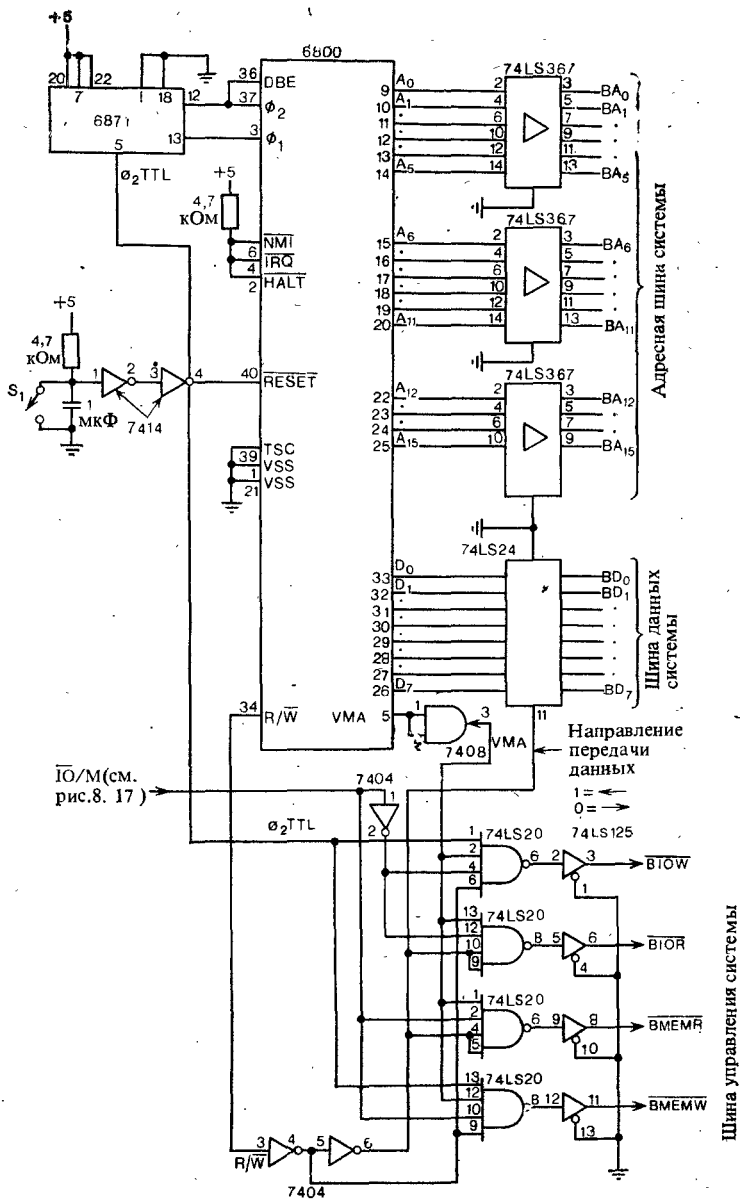
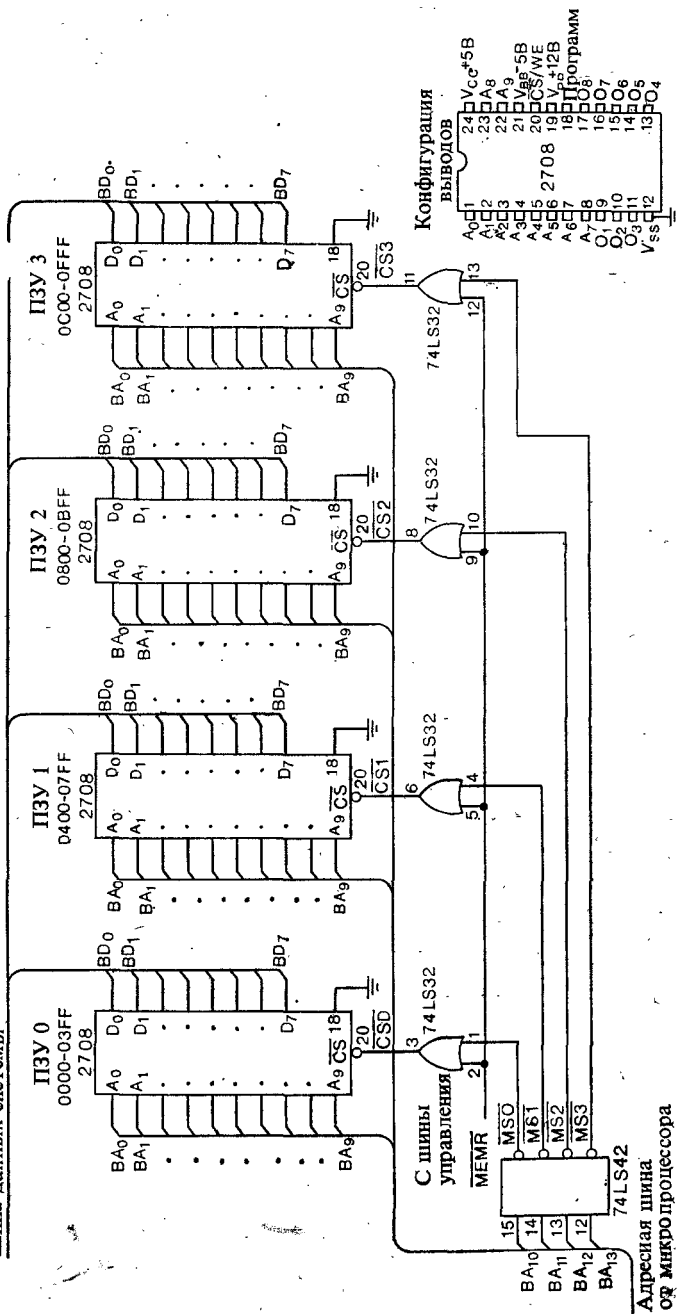


Рис. 8.16. Полная схема использования микропроцессора 6800 в качестве контроллера системы автоматического программирования ППЗУ.

Шина данных системы



использованием микропроцессоров четырех различных типов. Если читатель уяснил особенности взаимодействия технических средств системы с микропроцессором, то задача проектирования любой другой системы, реализующей другие функции, представит лишь еще один вариант использования тех же приемов и методов.

В гл. 9 будет рассмотрено программное обеспечение, предназначенное для управления техническими средствами, приведенными в данной главе. И в завершение в гл. 10 обсуждаются вопросы отладки системы в случае неправильного ее функционирования.

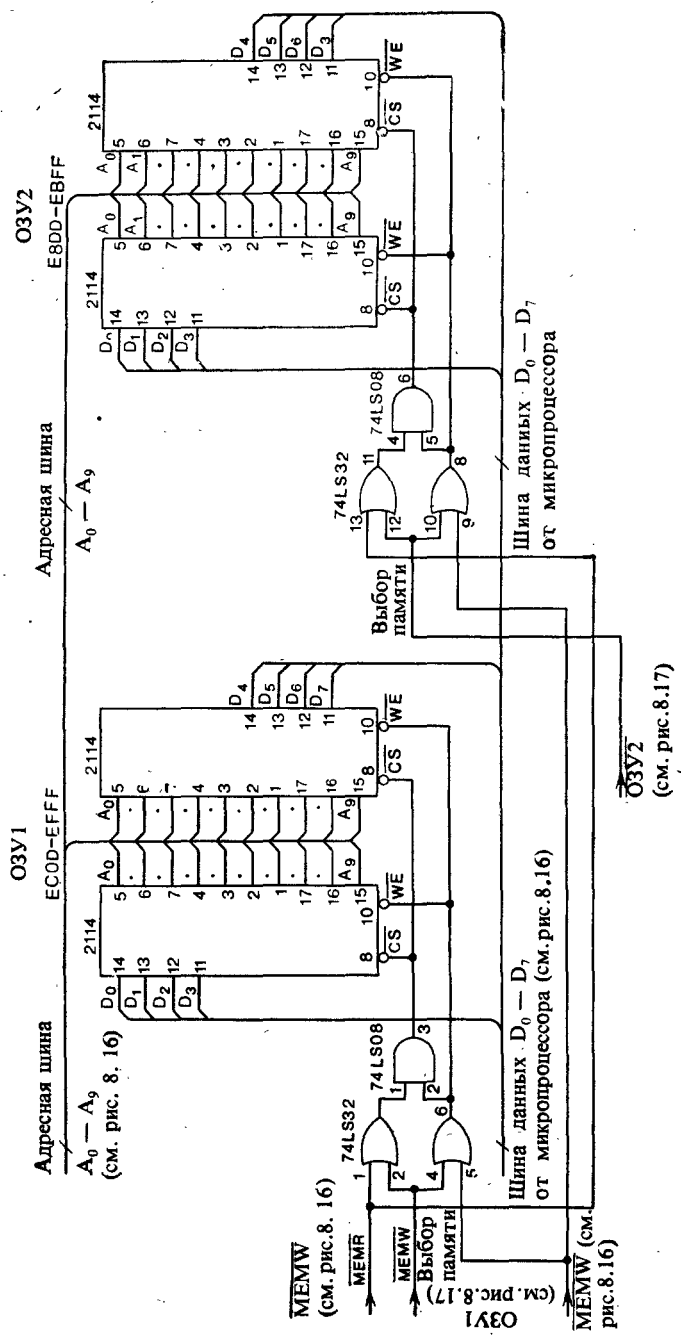


рис. 8.18. Полная схема интерфейса ОЗУ системы при использовании микропроцессора 6800 в качестве контроллера системы.

ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ УПРАВЛЕНИЯ МИКРОПРОЦЕССОРНОЙ СИСТЕМОЙ

В настоящей главе будет рассмотрено программное обеспечение, необходимое для управления техническими средствами системы, описанными в гл. 8. Каждая функция программного обеспечения будет проиллюстрирована блок-схемой. Для микропроцессоров 6800 и 8085 соответствующие блок-схемы раскрываются до уровня объектного кода. Для каждого микропроцессора приводятся полины распечатки программ.

После изучения предлагаемого материала читатель сможет самостоятельно разрабатывать программы управления микропроцессорной системой. Целью проводимого обсуждения является разбор конкретных решений по разработке программ путем установления взаимосвязи между окончательной реализацией программы и выполняемыми машиной функциями.

9.1. Начальный этап

Прежде чем мы сможем писать какие-либо программы, необходимо выяснить некоторые предварительные вопросы. Прежде всего необходимо выяснить, как программное обеспечение системы будет размещено; далее следует выяснить, как инициализируются технические средства или как они устанавливаются в начальное состояние при включении питания. Рассмотрим эти вопросы более подробно.

Структура размещения программного обеспечения представляет собой отображение всего адресного пространства и соответствующих функций каждого блока адресного пространства системы. Это первый уровень структуры размещения программного обеспечения. Размещение для системы, описанной в гл. 8, приведено на рис. 9.1. Как видно, рис. 9.1 дает точную картину адресов памяти и устройств ввода-вывода, используемых микропроцессором при установлении электрического соединения со всеми техническими средствами системы. При написании управляющих программ действует предположение, что технические средства работают правильно, так что любая ошибка в выполнении программы является собственной ошибкой программы, а не вызвана неисправностью оборудования.

При написании программ обычно не знают, как физически выглядят технические средства, особенности их функционирования учитываются на уровне адресов и двоичных разрядов. Имея эту информацию, можно правильно писать программы управления техническими средствами, не вникая в детали устройства последних.

Справедливым является и противоположный подход. При проектировании технических средств совсем не обязательно знать детали программного обеспечения. Можно так разрабо-

000—03FF	ПЗУ ₀	В этих блоках ПЗУ все управляющие программы
0400—07FF	ПЗУ ₁	
0800—0BFF	ПЗУ ₂	Пока не используются, предназначены для расширения системы
0C00—0FFF	ПЗУ ₃	Системное ОЗУ
1000—13FF	ОЗУ ₁	Память для 2708
1400—17FF	ОЗУ ₂	Не используется
1800—1BFF	ОЗУ ₃	

Ввод-вывод

F ₀ , F ₁ , F ₂	Индикатор
FE	Клавишный пульт
30, 31, 32, 33, 34, 35	Программная панель

Рис. 9.1. Распределение памяти системы, описанной в гл. 8.

тать технические средства, что микропроцессор будет осуществлять надежное взаимодействие со всеми элементами технических средств системы, не прибегая к помощи программного обеспечения. Если в оборудовании правильно выполнены соответствующие коммуникационные связи, программы будут использовать эти связи для реализации заданных функций.

Если же разработчик хорошо знает как программное обеспечение, так и технические средства, то результатом этого явится более простая и изящная структура системы в целом. Специалист по техническим средствам, например, может спроектировать систему, ориентируясь на особенности программного обеспечения конкретного микропроцессора. Систему можно также создавать и таким образом, чтобы специфические особенности программных средств конкретного микропроцессора не оказывали существенного влияния. С другой стороны, если программист знает технические средства, он может создавать такие программы, которые наиболее полно используют их возможности. Если же в разработке системы участвуют два специалиста, то наилучшие результаты достигаются тогда, когда программист и разработчик аппаратуры тесно взаимодействуют друг с другом. В этом случае каждый из них будет знать намерения другого. Это часто позволяет достичь значительной экономии усилий путем рационального распределения функций между техническими средствами и программным обеспечением,

что невозможно при отсутствии подобного сотрудничества разработчиков.

В идеальном случае разработчик технических средств либо программного обеспечения микропроцессорных систем должен знать обе эти области настолько, чтобы суметь оценить возникающие проблемы и установить эффективное взаимодействие между техническими и программными средствами.

В технике существует отчетливая граница между техническими средствами и программным обеспечением. В ряде ситуаций соблюдение этой границы является необходимым при разработке систем. Однако для проектировщиков систем на базе микропроцессоров, как инженеров, так и техников, знание технических средств и программного обеспечения одновременно является чрезвычайно ценным.

Возвращаясь к распределению памяти рассматриваемой системы, показанному на рис. 9.1, можно заметить отсутствие некоторых деталей, относящихся к разрядам данных D0—D7, с помощью которых осуществляется управление техническими средствами. В случае ОЗУ и ПЗУ системы это видно непосредственно. Так, выходы O_1 — O_8 ПЗУ 2708 соединяются соответственно с D0—D7. Аналогичное справедливо и для ОЗУ системы. Не имеет значения, какие разряды данных ОЗУ поступают на конкретные линии шины данных системы, поскольку при записи данных в ОЗУ и при их считывании из ОЗУ сохраняется соответствие между разрядами данных и линиями шины данных. Это означает, что в случае ОЗУ и ПЗУ конкретная привязка разрядов данных к линиям шины безразлична. Однако этот вопрос становится важным при работе с устройствами ввода-вывода.

Рассмотрим, например, случай, когда микропроцессор соединен с устройством отображения. В этой ситуации программисту необходимо точно знать соответствие разрядов данных входа устройства, чтобы осуществлять выдачу на табло конкретных цифр. Подобная информация обеспечивается техническими средствами системы. На рис. 9.2 показано конкретное назначение разрядов данных для каждого устройства ввода-вывода системы, что снимает неопределенность, допущенную в предыдущем разделе.

Из рис. 9.2 видно, что для каждого порта ввода и вывода определена точная функция каждого разряда данных. Сбрасываемое разрядов, не описанных для конкретного устройства ввода-вывода, не используется и может быть произвольным. В документации на систему информация, приведенная на рис. 9.2, может быть представлена самым различным образом. Однако несмотря на форму представления, наличие этой специфической информации необходимо до начала разработки программного обеспечения.

АДРЕСНОЕ ПРОСТРАН- СТВО

АДРЕСНОЕ ПРОСТРАН- СТВО	Разряды дан- ных	Функция
FE (вывод)	D ₀ D ₁ D ₂ D ₃ D ₄ D ₅ , D ₆ , D ₇	R ₀ R ₁ R ₂ R ₃ R ₄ Не используется
FE (ввод)	D ₀ D ₁ D ₂ D ₃ D ₄ D ₅ , D ₆ , D ₇	C ₀ C ₁ C ₂ C ₃ C ₄ Не используется
F ₂ (вывод)	D ₇ D ₆ D ₅ D ₄ D ₃ D ₂ D ₁ D ₀	MSD (8-й разряд) MSD (4-й разряд) MSD (2-й разряд) MSD (1-й разряд) (MSD - 1) (8-й разряд) (MSD - 1) (4-й разряд) (MSD - 1) (2-й разряд) (MSD - 1) (1-й разряд)
F ₁ (вывод)	D ₇ D ₆ D ₅ D ₄ D ₃ D ₂ D ₁ D ₀	(MSD - 2) (8-й разряд) (MSD - 2) (4-й разряд) (MSD - 2) (2-й разряд) (MSD - 2) (1-й разряд) (LSD + 2) (8-й разряд) (LSD + 2) (4-й разряд) (LSD + 2) (2-й разряд) (LSD + 2) (1-й разряд)
F ₀ (вывод)	D ₇ D ₆ D ₅ D ₄ D ₃ D ₂ D ₁ D ₀	(LSD + 1) (8-й разряд) (LSD + 1) (4-й разряд) (LSD + 1) (2-й разряд) (LSD + 1) (1-й разряд) ESD (8-й разряд) LSD (4-й разряд) LSD (2-й разряд) LSD (1-й разряд)
30 (вывод)	D ₀ —D ₇	Данные для программирования ППЗУ
31 (ввод)	D ₀ —D ₇	Данные с выходов ППЗУ
32 (вывод)	D ₀ = 1 D ₀ = 0 D ₁ —D ₇	Чтение данных из ППЗУ Запись данных в ППЗУ Не используется
33 (вывод)	D ₀ —D ₇	PA ₀ —PA ₇ , адрес ППЗУ
34 (вывод)	D ₀ , D ₁ D ₂ —D ₇	PA ₈ —PA ₉ , адрес ППЗУ Не используется
35 (вывод)	D ₁ = 1 D ₀ = 0 D ₁ = 1 D ₁ = 0 D ₂ —D ₇	CS/WE = +12 В CS = 0,0 В Импульс программирования, = +26 В Импульс программирования 0,0 В

Рис. 9.2. Таблица функционального назначения разрядов для всех портов ввода и вывода системы. Для некоторых портов отмечено наличие неиспользуемых разрядов.

Кроме того, прежде чем приступить к написанию программ, необходимо знать, в какое состояние устанавливаются (как инициализируются) конкретные устройства ввода-вывода при подаче питания в систему. Обычно указатель стека системы всегда необходимо устанавливать в начальное состояние. Кроме того, если в системе используются прерывания, то возникает необходимость в установлении масок прерываний и разрешения либо запрещения функции прерывания. После того как выполнены подобные общие действия по инициализации, следует обратиться к специальной информации, необходимой для инициализации остальных технических средств системы.

Для рассматриваемой системы в этой связи характерно следующее:

1. Импульс программирования — 0 В.
2. Напряжение CS/WE—0 В.
3. Для порта управления 32_{16} D0=1. Это открывает передачу данных из ППЗУ в систему (см. рис. 9.2).
4. Устройство отображения системы устанавливается в F00000.
5. В системе не используются прерывания.
6. Указатель стека устанавливается 03FF₁₆ (в соответствии с распределением памяти, приведенным на рис. 9.1).

Таковы основные действия по инициализации технических средств, необходимые для рассматриваемой системы. Возможно, потребуется инициализация программных средств, связанная с присвоением определенных значений некоторым переменным. Это станет ясно при написании программ. Теперь имеется вся информация, необходимая для того, чтобы приступить к программированию.

9.2. Общее представление о программном обеспечении

Прежде чем рассматривать частные вопросы написания программ, проанализируем общую структуру программного обеспечения системы. Поскольку система взаимодействует с клавишным пультом, то, прежде чем начать выполнение некоторой подпрограммы, она ждет ввода определенного ключа. Только после ввода ключа, основываясь на функции, задаваемой этим ключом, система выполняет определенную программу. По завершении программы снова наступает состояние ожидания ввода очередного ключа.

Чтобы заставить пользователя вводить правильную информацию, система должна дать ему «подсказку» с помощью табло. Именно с помощью такого «интеллектуального» использования клавишного пульта и табло и реализуются функции, перечисленные для системы, рассматриваемой в гл. 8.

Общая блок-схема программного обеспечения системы приведена на рис. 9.3, который поясняет последовательность работ при написании программ.

Рис. 9.3. Укрупненная блок-схема функционирования системы, отражающая основные выполняемые операции. Содержание каждого блока подробно раскрывается в главе.



9.3. Программы ввода информации с клавишного пульта

Функционирование системы зависит от ее способности различать конкретный ключ из числа ключей, формирующих клавишный пульт устройства ввода. В гл. 4 была рассмотрена программная реализация этой функции. Рассмотрим еще раз подпрограмму ввода с клавишного пульта, приведенную в гл. 4. Эта подпрограмма для удобства еще раз повторена на рис. 9.4. Отметим, что значения переменных, установленные подпрограммой рис. 9.4, являются десятичными числами, определяющими вес ключа в матрице пульта. Функция подпрограммы может быть представлена в виде блок-схемы, приведенной на рис. 9.5.

Согласно рис. 9.5, выход из подпрограммы ввода ключа в основную программу не происходит до тех пор, пока не определен требуемый ключ и не записан его десятичный вес в ячейку памяти. На рис. 9.6 приведены десятичные веса для всех ключей матрицы пульта. Видно, что после выполнения подпрограммы ввода ключа в ячейку KWGT заносится целое число, принимающее значения от 0 до 24 включительно.

Напомним, что функции ключей матрицы пока еще не определены. Определим эти функции так, как показано на рис. 9.7. Отметим, что для всех ключей данное число в ячейке KWGT соответствует значению ключа. Это означает, что при

```

1 0000 *****
2 0000 *
3 0000 *   ПРОГРАММА ВВОДА ДАННЫХ С ПУЛЬТА И ВЫВОДА НА ДИСПЛЕЙ
4 0000 *
5 0000 ***** ПРОГРАММИСТ ДЖИМ КОФФРОН 8-28-79 *****
6 0000 *
7 0000 *
8 0000   ORG 00   УСТАНОВИТЬ НАЧ.АДРЕС ПРОГРАММЫ
9 0000 *
10 0000 *   ВО-ПЕРВЫХ, НЕОБХОДИМО ИНИЦИАЛИЗИРОВАТЬ ВСЕ ПЕРЕМЕННЫЕ
11 0000 *
12 0000 31 FF 13   LXI SP, 13FFH   ИНИЦИАЛИЗИРОВАТЬ УКАЗАТЕЛЬ СТЕКА
13 0003 AF        XRA A           НАЧ.ЗНАЧЕНИЯ ФЛАЖКОВ, ОБНУЛЕНИЕ АККУМ
14 0004 FB        EI             РАЗРЕШИТЬ ПЕРЕРЫВАНИЯ
15 0005 D3 F0     OUT OF0H        ОБНУЛИТЬ ДИСПЛЕЙ
16 0007 D3 F1     OUT OF1H
17 0009 D3 F2     OUT OF2H        ВЫДАТЬ НА ДИСПЛЕЙ 000000
18 000B 32 00 10   STA KTIME     KTIME=0
19 000E 32 01 10   STA CFLAG     CFLAG=0
20 0011 32 02 10   STA KWGHT     KWGHT=0
21 0014 32 03 10   STA KCOMP     KCOMP=0
22 0017 3C        INR A           АККУМУЛЯТОР=1
23 0018 32 04 10   STA KROW      KROW=00000001
24 001B *
25 001B *
26 001B *   УСТАНОВИТЬ АДРЕСА ПЕРЕМЕННЫХ
27 001B *
28 001B   KTIME EQU 1000H
29 001B   CFLAG EQU KTIME+1,
30 001B   KWGHT EQU CFLAG+1
31 001B   KCOMP EQU KWGHT+1
32 001B   KROW EQU KCOMP+1
33 001B   NROW EQU KROW+1
34 001B *
35 001B *
36 001B ***** НАЧАЛО ПРОГРАММЫ *****
37 001B *
38 001B *   НАЧАЛО РИС.4.20
39 001B *
40 001B CD EB 00   SROW CALL JROW   ВЫВЕСТИ АКТИВНЫЙ НАБОР
41 001E CD DB 00   CALL COLM       ВЫЗВАТЬ ПОДПРОГРАММУ COLM
42 0021 3A 01 10   LDA CFLAG       НЕОБХОДИМО ПРОВЕРИТЬ ФЛАЖОК СТОЛБЦА
43 0024 FE 00     CPI 00H          ПРОВЕРИТЬ АКТИВНОСТЬ ЦИКЛА
44 0026 CA 1B 00   JZ SROW         НЕ АКТИВНЫЙ СТОЛБЕЦ, ПОВТОРИТЬ ЦИКЛ
45 0029 CD B1 00   CALL KEYW       АКТИВНЫЙ СТОЛБЕЦ, КАКОЕ ИМЕННО???
46 002C *
47 002C *   КОНЕЦ РИС.4.20
48 003C *
49 002C *   НАЧАЛО РИС.4.21
50 002C *
51 002C ***** БЫЛИ НАЖАТЫ КЛАВИШИ *****
52 002C *
53 002C ***** (ШАГ 1)
54 002C *
55 002C 3A 00 10   LDA KTIME       ПЕРЕЗАГРУЗИТЬ KTIME ИЗ ПАМЯТИ
56 002F *
57 002F *
58 002F ***** (ШАГ 2)
59 002F *
60 002F FE 00     CPI 00H          KTIME=0??

```

Рис. 9.4. Программа ввода ключа и вывода информации на табло для микропроцессора 8080.

```

61 0031 CA 5B 00      JZ   KCLO1      ДА, ЭТО ПЕРВЫЙ РАЗ
62 0034      *
63 0034      * ***** НЕ ПЕРВЫЙ РАЗ *****
64 0034      *
65 0034      * ***** (ШАГ 7)
66 0034      *
67 0034 3A 02 10      LDA   KWGHT      КАКАЯ ЦИФРА БЫЛА НАВРАНА???
68 0037 4F            MOV   C, A      РЕГИСТР C=KWGHT
69 0038 3A 03 10      LDA   KCOMP      РЕГИСТР A=KCOMP
70 0038 B9            CMP   C      KCOMP=KWGHT???
71 003C CA 46 00      JZ     KCLO2      ДА, ОНИ РАВНЫ!!!
72 003F      *
73 003F      * ***** (ШАГ 8)
74 003F      *
75 003F AF            XRA   A      ОНИ НЕ РАВНЫ
76 0040 32 00 10      STA   KTIME      ПЕРЕЗАПИСАТЬ KTIME
77 0043 C3 66 00      JMP   KCLO3      ИДТИ НА ПЕРЕЗАГРУЗКУ АКТИВНОГО НАБОРА
78 0046      *
79 0046      * ***** (ШАГ 9)
80 0046      *
81 0046 3A 00 10      KCLO2 LDA   KTIME      ВЫЗВАТЬ KTIME ИЗ ПАМЯТИ
82 0049 3C            INR   A      KTIME=KTIME+1
83 004A 32 00 10      STA   KTIME      KTIME=KTIME+1
84 004D      *
85 004D      * ***** (ШАГ 10)
86 004D      *
87 004D FE 32            CPI   50      KTIME=50???
88 004F C2 66 00      JNZ   KCLO3      ЕЩЕ НЕ 50
89 0052 CD 6E 00      CALL  KOPN      ПРОВЕРИТЬ ГОТОВНОСТЬ ПУЛЬТА
90 0055 CD A7 00      CALL  KOUT      ВЫДАТЬ НА ПУЛЬТ
91 0058 C3 1B 00      JMP   SROW      НАЗАД К НАЧАЛУ
92 005B      *
93 005B      * ***** (ШАГ 3)
94 005B      *
95 005B 3A 02 10      KCLO1 LDA   KWGHT
96 005E 32 03 10      STA   KCOMP      KCOMP=KWGHT
97 0061      *
98 0061      * ***** (ШАГ 4)
99 0061      *
100 0061 3E 01            MVI   A, 01
101 0063 32 00 10      STA   KTIME      УСТАНОВИТЬ KTIME=1
102 0066      *
103 0066      * ***** (ШАГ 5)
104 0066      *
105 0066 3E 01      KCLO3 MVI   A, 01
106 0068 32 04 10      STA   KROW      УСТАНОВИТЬ АКТИВНЫЙ НАБОР=00000001
107 006B      *
108 006B      * ***** (ШАГ 6)
109 006B      *
110 006B C3 1B 00      JMP   SROW      ИДТИ К НАЧАЛУ ПРОГРАММЫ
111 006E      *
112 006E      *
113 006E      * ***** НАЧАЛО ПОДПРОГРАММЫ *****
114 006E      *
115 006E      * *****
116 006E      * *****
117 006E      *
118 006E      *
119 006E      *
120 006E      * РИС.4.22

```

```

121 006E      *   ПРОГРАММА ПОДГОТОВКИ ПУЛЬТА
122 006E      *
123 006E      *****
124 006E      *
125 006E      ***** (ШАГ 1)
126 006E      *
127 006E 3E 01 KOPN   MVI   A,01
128 0070 32 04 10 STA   KROW   KROW=00000001
129 0073 3D      DSR   A
130 0074 32 05 10 STA   NROW   NROW=00000000
131 0077      *
132 0077      ***** (ШАГ 2)
133 0077      *
134 0077 32 00 10 STA   KTIME   KTIME=0
135 007A      *
136 007A      ***** (ШАГ 3)
137 007A      *
138 007A CD EB 0Q KOPN1 CALL  OROW   ВЫВЕСТИ НАБОР
139 007D      *
140 007D      ***** (ШАГ 4)
141 007D      *
142 007D CD DB 00      CALL  COLM   ВВЕСТИ ДАННЫЕ СТОЛБЦА
143 0080      *
144 0080      ***** (ШАГ 5)
145 0080      *
146 0080 3A 01 10      LDA   CFLAG   ВВЕСТИ CFLAG
147 0083 FE 00      CPI   00   CFLAG=0???
148 0085 C2 6E 00      JNZ   KOPN   ПУЛЬТ ЕЩЕ НЕ СВОБОДЕН
149 0088      *
150 0088      ***** (ШАГ 9)
151 0088      *
152 0088 3A 05 10      LDA   NROW   ВВЕСТИ АКТИВНЫЙ НАБОР
153 008B 3C      INR   A
154 008C 32 05 10      STA   NROW   ЗАПИСАТЬ АКТИВНЫЙ НАБОР
155 008F      *
156 008F      ***** (ШАГ 10)
157 008F      *
158 008F FE 05      CPI   05
159 0091 C2 7A 00      JNZ   KOPN1   СКАНИРОВАНИЕ НЕ ЗАКОНЧЕНО
160 0094      *
161 0094      ***** (ШАГ 6)
162 0094      *
163 0094 3A 00 10      LDA   KTIME
164 0097 3C      INR   A
165 0098 32 00 10      STA   KTIME
166 009B 47      MOV   B,A   РЕГИСТР В=KTIME
167 009C      *
168 009C      ***** (ШАГ 6A)
169 009C      *
170 009C AF      XRA   A
171 009D 32 05 10      STA   NROW   ЗНАЧЕНИЕ НАБОРА=0
172 00A0      *
173 00A0      ***** (ШАГ 7)
174 00A0      *
175 00A0 3E 32      MVI   A,50
176 00A2 B8      CMP   B   KTIME=50???
177 00A3 C2 7A 00      JNZ   KOPN1   НЕТ, СКАНИРОВАНИЕ ПОВТОРИТЬ
178 00A6      *
179 00A6      ***** (ШАГ 8)
180 00A6      *

```

```

181 00A6 C9          RET          ВЫХОД ИЗ ПОДПРОГРАММЫ ПОДГОТОВКИ
182 00A7          *
183 00A7          *
184 00A7          *****
185 00A7          *
186 00A7          *      ПОДПРОГРАММА KOUT
187 00A7          *
188 00A7          *      ЭТА ПОДПРОГРАММА ВЫВЕДЕТ ДАННЫЕ С ПУЛЬТА НА ДИСПЛЕЙ
189 00A7          *
190 00A7          *****
191 00A7          *
192 00A7 3A 02 10 KOUT LDA KWCHT    ДАННЫЕ С ПУЛЬТА В АККУМУЛЯТОРЕ
193 00AA D3 F0      OUT OF0H
194 00AC D3 F1      OUT OF1H
195 00AE D3 F2      OUT OF2H
196 00B0 C9          RET
197 00B1          *
198 00B1          *****
199 00B1          *
200 00B1          *      ПОДПРОГРАММА KEYW
201 00B1          *
202 00B1          *      ЭТА ПОДПРОГРАММА УСТАНОВИТ ЗНАЧЕНИЕ НАЖАТОЙ КЛАВИШИ
203 00B1          *
204 00B1          *****
205 00B1          *
206 00B1          *      ***** БЛОК-СХЕМА ЭТОЙ ПРОГРАММЫ НА РИС.4.18
207 00B1          *
208 00B1          *      ***** (ШАГ 1)
209 00B1          *
210 00B1 AF KEYW XRA A      ОБНУЛИТЬ ФЛАЖКИ И АККУМУЛЯТОР
211 00B2 3A 01 10 LDA CFLAG    ПРИНЯТЬ ФЛАЖОК СТОЛБЦА ИЗ ПАМЯТИ
212 00B5          *
213 00B5          *      ***** (ШАГ 2)
214 00B5          *
215 00B5 06 00      MVI B,00    ОБНУЛИТЬ СЧЕТЧИК
216 00B7          *
217 00B7          *      ***** (ШАГ 3)
218 00B7          *
219 00B7 1F KEYW1 RAR      СДВИНУТЬ ВПРАВО НА 1 БИТ, 0 В D7
220 00B8 FE 00      CPI 00
221 00BA          *
222 00BA          *      ***** (ШАГ 4)
223 00BA          *
224 00BA CA C1 00      JZ KEYW2    РЕГИСТР B=0,1,2,3,4 (ШАГ 6)
225 00BD          *
226 00BD          *      ***** (ШАГ 5)
227 00BD          *
228 00BD 04      INR B
229 00BE C3 B7 00      JMP KEYW1    УВЕЛИЧИТЬ ЗНАЧ.СЧЕТЧИКА И ИДТИ НА ПОВ.
230 00C1          *
231 00C1          *      ***** (ШАГ 7)
232 00C1          *
233 00C1 AF KEYW2 XRA A      ОБНУЛИТЬ ФЛАЖКИ
234 00C2 3A 04 10 LDA KROW    ВЫЗВАТЬ АКТИВНЫЙ НАБОР
235 00C5          *
236 00C5          *      ***** (ШАГ 8)
237 00C5          *
238 00C5 0E 00      MVI C,00    ОБНУЛИТЬ СЧЕТЧИК T2
239 00C7          *
240 00C7          *      ***** (ШАГ 9)

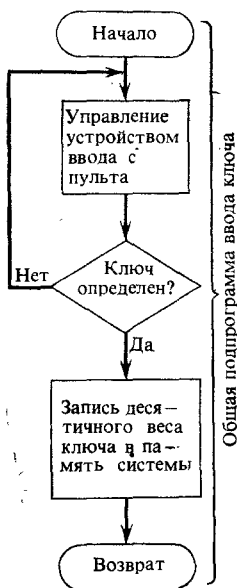
```


241	00C7	*		
242	00C7 1F	KEYW3	RAR	СДВИГ ВПРАВО НА 1 ВИТ, 0 В D7
243	00C8 FE 00		CPI 00	
244	00CA	*		
245	00CA	*****	(ШАГ 10)	
246	00CA	*		
247	00CA CA D1 00		JZ KEYW4	ЕСЛИ 0, ТО ВЫПОЛНЕНО
248	00CD	*		
249	00CD	*****	(ШАГ 14)	
250	00CD	*		
251	00CD 0C		INR C	УВЕЛИЧИТЬ ЗНАЧЕНИЕ СЧЕТЧИКА
252	00CE C3 C7 00		JMP KEYW3	ИДТИ НА ПОВТОР
253	00D1	*		
254	00D1	*****	(ШАГ 11)	
255	00D1	*		
256	00D1 AF	KEYW4	XRA A	ОБНУЛИТЬ ФЛАЖКИ
257	00D2 78		MOV A, B	АККУМУЛЯТОР=СЧЕТЧИК СТОЛБЦОВ (T1)
258	00D3 17		MOV A, B	B*2
259	00D4 17		RAL	B*4
260	00D5 80		ADD B	B*5
261	00D6 81		ADD C	(B*5)+C=0-24
262	00D7	*		
263	00D7	*****	(ШАГ 12)	
264	00D7	*		
265	00D7 32 02 10		STA KWGHT	
266	00DA	*		
267	00DA	*****	(ШАГ 13)	
268	00DA	*		
269	00DA C9		RET	ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
270	00DB	*		
271	00DB	*****		
272	00DB	*		
273	00DB	*	ПОДПРОГРАММА COLM	
274	00DB	*		
275	00DB	*	ЭТА ПОДПРОГРАММА ОПРЕДЕЛИТ АКТИВНЫЙ СТОЛБЕЦ,	
276	00DB	*	ЕСЛИ ОН СУЩЕСТВУЕТ	
277	00DB	*		
278	00DB	*****		
279	00DB	*		
280	00DB	*	ЭТА ПОДПРОГРАММА-РИС.4.15	
281	00DB	*		
282	00DB	*****		
283	00DB	*		
284	00DB	*****	(ШАГ 1)	
285	00DB	*		
286	00DB AF	COLM	XRA A	
287	00DC 32 01 10		STA CFLAG	УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА=0
288	00DF	*		
289	00DF	*****	(ШАГ 2)	
290	00DF	*		
291	00DF DB FE		IN OFEH	ВХОДНЫЕ ДАННЫЕ ИЗ ПОРТА FE
292	00E1	*		
293	00E1	*****	(ШАГ 3)	
294	00E1	*		
295	00E1 F6 E0		ORI OEON	УСТАНОВИТЬ БИТЫ D5, D6, D7=1
296	00E3	*		
297	00E3	*****	(ШАГ 4)	
298	00E3	*		
299	00E3 FE FF		CPI OFEH	
300	00E5 C8		RZ	ШАГ (4A)

```

301 00E6      *
302 00E6      ***** (ШАГ 5)
303 00E6      *
304 00E6 2F      CMA                ИНВЕРТИРОВАТЬ ДАННЫЕ
305 00E7      *
306 00E7      ***** (ШАГ 6)
307 00E7      *
308 00E7 32 01 10 STA CFLAG      УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА
309 00EA C9      RET                ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
310 00EB      *
311 00EB      *****
312 00EB      *
313 00EB      * ПОДПРОГРАММА ВЫВОДА АКТИВНОГО НАБОРА НА ПУЛЬТ
314 00EB      *
315 00EB      * ИМЯ ЭТОЙ ПОДПРОГРАММЫ - OROW
316 00EB      *
317 00EB      *
318 00EB      *
319 00EB      *****
320 00EB      *****
321 00EB      *
322 00EB      * ПОДПРОГРАММА К РИС.4.10 В МНЕМОНИКЕ 8080
323 00EB      *
324 00EB      *****
325 00EB      *
326 00EB      *
327 00EB 3A 04 10 OROW LDA KROW      ПОВТОРНЫЙ ВЫЗОВ АКТИВНОГО НАБОРА
328 00EE      *
329 00EE      ***** (ШАГ 2)
330 00EE      *
331 00EE FE 10      CPI 10H          СКАНИРОВАНИЕ НАБОРА ЗАКОНЧЕНО???
332 00F0 C2 F8 00  JNZ OROW1        ЕСЛИ НЕТ, ТО ШАГ 2B
333 00F3      *
334 00F3      ***** (ШАГ 2A) -
335 00F3      *
336 00F3 3E 01      MVI A, 01H       ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАБОР
337 00F5 C3 F9 00  JMP ST3          ИДТИ НА ШАГ 3
338 00F8      *
339 00F8      ***** (ШАГ 2B)
340 00F8      *
341 00F8 07      OROW1 RLC           СДВИНУТЬ ДАННЫЕ ВЛЕВО. 0 В DO
342 00F9      *
343 00F9      ***** (ШАГ 3)
344 00F9      *
345 00F9 32 04 10 ST3 STA KROW      СОХРАНИТЬ АКТИВНЫЙ НАБОР
346 00FC      *
347 00FC      ***** (ШАГ 4)
348 00FC      *
349 00FC 2F      CMA                ИНВЕРТИРОВАТЬ СЛОВО НАБОРА
350 00FD      *
351 00FD      ***** (ШАГ 5)
352 00FD      *
353 00FD D3 FE      OUT OFEH        ВЫВЕСТИ АКТИВНЫЙ НАБОР В ПОРТ FE
354 00FF      *
355 00FF      ***** (ШАГ 6)
356 00FF      *
357 00FF C9      RET
358 0100      *
360 0100      *****

```



	C ₀	C ₁	C ₂	C ₃	C ₄
R ₀	X 0	X 1	X 2	X 3	X 4
R ₁	X 5	X 6	X 7	X 8	X 9
R ₂	X 10	X 11	X 12	X 13	X 14
R ₃	X 15	X 16	X 17	X 18	X 19
R ₄	X 20	X 21	X 22	X 23	X 24

Рис. 9.6. Аналогичен рис. 4.17. Показывает десятичный вес каждого переключателя матрицы пульта размером 5×5 .

Рис. 9.5. Блок-схема, демонстрирующая реализацию функции ввода ключа с клавишного пульта. Данная последовательность действий повторяется каждый раз при выполнении программы рис. 9.4

Функция	Вес ключа, вычисляемый подпрограммой ввода ключа
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15
Ввод	16
Стирание ввода	17
Установка адреса	18
Стирание ОЗУ	19
Программирование	20
Верификация	21
Копирование	22
Не используется	23, 24

Рис. 9.7. Функции ключей (левая колонка) и их веса (правая колонка).

нажатии на пульте ключа «В» в KWGT записывается двоичное число 1011, представляющее значение десятичного 11 (B_{16}). Таким образом, при программной реализации процедуры нет необходимости вычислять двоичное значение ключа данных, поскольку это значение автоматически записывается в ячейку.

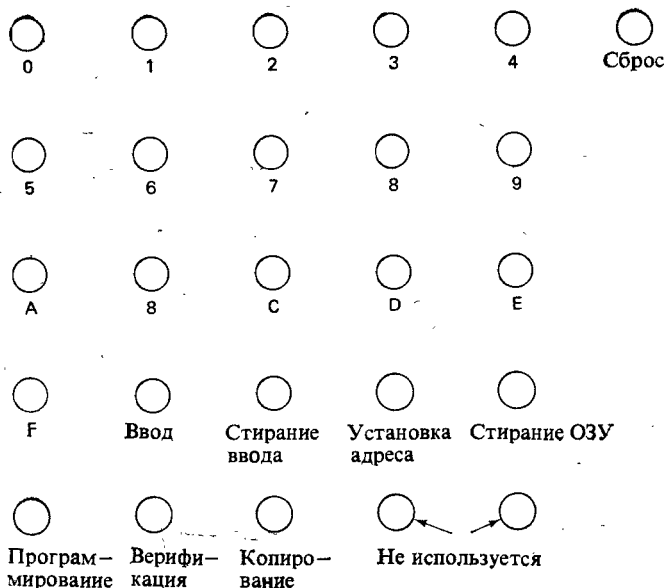


Рис. 9.8. Пространственное расположение клавиатуры с пометкой функций ключей. Ключ сброса не является частью матрицы пульта. Этот ключ играет особую роль и используется просто для сброса системы.

В соответствии с определением ключей, приведенным на рис. 9.7, физическая реализация клавишного пульта будет соответствовать представленной на рис. 9.8. Здесь ключ сброса не входит в матрицу ключей. Этот ключ играет особую роль, осуществляя немедленный сброс системы вне зависимости от выполняемой ею функции.

На рис. 9.4 были приведены программные средства микропроцессора 8085, необходимые для реализации функции подпрограммы KEYIN. Эти средства подробно рассматривались в гл. 4, так что любые неясности рис. 9.4 можно устранить, используя материалы этой главы.

9.4. Главная управляющая программа

Обратимся теперь к особенностям построения главной управляющей программы. Эта программа управляет работой системы на основе значения введенного ключа. После реализации каждой частной функции системы управление возвращается в

главную управляющую программу. На рис. 9.3 была показана укрупненная блок-схема этой основной операционной программы. На рис. 9.9 представлена более подробная блок-схема этой программы.

Видно, что одним из первых действий главной управляющей программы является вызов подпрограммы ввода ключа — шаг 1а. Вспомним, что система выполняет эту подпрограмму до тех пор, пока не будет введено требуемое значение ключа. После выхода из подпрограммы ввода ключа производится проверка значения ключа, записанного в ячейке KWGT. Если это значение соответствует допустимой функции, осуществляется переход к шагу 5.

Система ожидает ввода функционального ключа, а после осуществления этой операции ей безразлично конкретное значение ключа. По значениям функциональных ключей видно, что любое число в ячейке KWGT, большее чем 15, определяет допустимую функцию. Поэтому на шаге 2 блок-схемы рис. 9.9 осуществляется проверка логического условия, является ли число в ячейке KWGT больше 15.

Если же число в KWGT не больше 15, на табло высвечивается системная ошибка. Этой частной ошибке приписан номер 1, так что на табло заносится число E00001₁₆. Это число, определяющее факт ввода недопустимого ключа (ошибки), высвечивается до тех пор, пока не будет нажат ключ сброса. Поскольку в процессе написания программного обеспечения генерируются коды других ошибок, необходимо привести и включить в документацию список кодов ошибок и их смысловые значения.

На шаге 5 рис. 9.9 вычисляется адрес программного модуля, который осуществляет управление системой при реализации заданной функции. Вычисление адреса производится следующим образом. Прежде всего подготовим таблицу операторов перехода, содержащую обращение к меткам системы, как это показано на рис. 9.10. Адрес, определяемый на основе числа, записанного в ячейку KWGT, указывает на один из операторов перехода, приведенных на рис. 9.10; этот оператор позволяет осуществить передачу управления в ячейку, с которой начинается частный модуль программного обеспечения, соответствующий введенному ключу.

Оператор перехода рис. 9.10, выбираемый для выполнения, соотносится конкретному значению числа в KWGT. Например, если в эту ячейку записано число 16₁₀, то будет выполнен оператор JMP ENTER; в случае числа 17₁₀ — оператор JMP CENTY. Отметим, что каждый раз при увеличении на единицу числа в KWGT, получаемого в результате выполнения подпрограммы KEYIN, осуществляется выбор следующего по порядку оператора перехода на рис. 9.10.

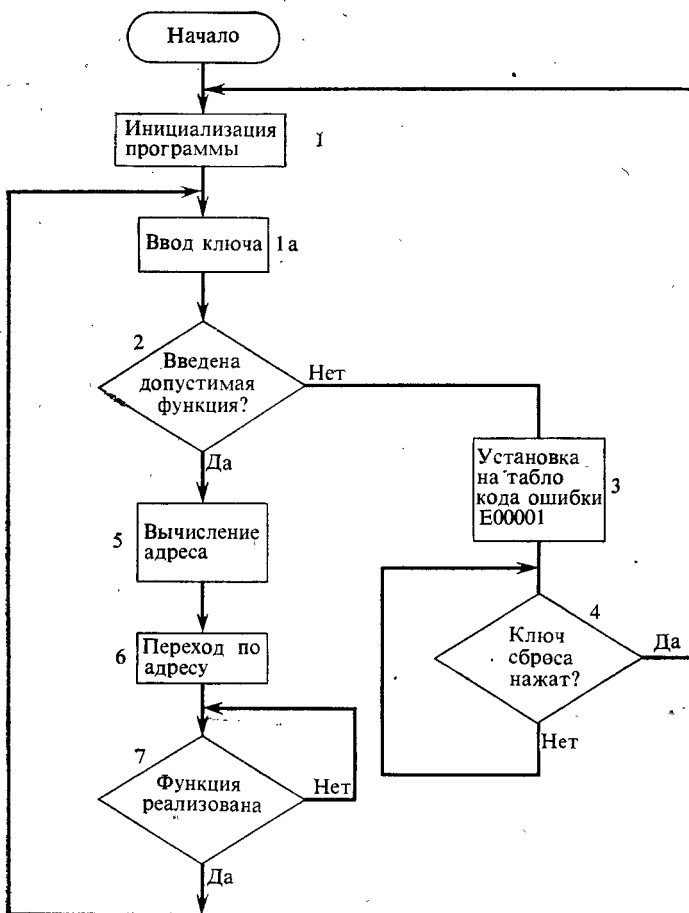


Рис. 9.9. Блок-схема главной управляющей программы системы.

Таблица	JMP	ВВОД
	JMP	Стирание ВВОДА
	JMP	Установка АДРЕСА
	JMP	Стирание ОЗУ
	JMP	Программирование
	JMP	Верификация
	JMP	Копирование
	JMP	NA ₁
	JMP	NA ₂

Рис. 9.10. Таблица операторов перехода, указывающих модули программы, реализующие отмеченные функции.

Поэтому требуемый оператор JMP относительно первого оператора может быть вычислен следующим образом:

JMP относительно JMP ENTER = KWGT — 16₁₀.

Для иллюстрации этой формулы положим, что в ячейке KWGT записано число 19₁₀, сформированное подпрограммой KEYIN, что соответствует ключу СТИРАНИЕ ОЗУ. По приведенной формуле вычисляется следующий оператор перехода:

JMP относительно JMP ENTER = KWGT — 16 = 19 — 16 = 3.

Таким образом, выполняется третий по порядку (от JMP ENTER) оператор перехода. Выполнение этого оператора при-

7 0000	*	
8 0100		ORG 0100H УСТАНОВИТЬ НАЧАЛО ТАБЛИЦЫ
9 0100	*	
10 0100	****	НАЧАЛО ТАБЛИЦЫ *****
11 0100	*	
12 0100 C3 33 OF		JMP ENTER
13 0103 C3 34 OF		JMP CENTY
14 0106 C3 35 OF		JMP ADDSE
15 0109 C3 36 OF		JMP CLRAM
16 010C C3 37 OF		JMP PROG
17 010F C3 38 OF		JMP VEY
18 0112 C3 39 OF		JMP COPY
19 0115 C3 3A OF		JMP NA1
20 0118 C3 3B OF		JMP NA2
21 011B	*	
22 011B	****	КОНЕЦ ТАБЛИЦЫ *****
23 011B	*	

Рис. 9.11. Таблица операторов перехода рис. 9.10 после трансляции. Начало таблицы соответствует адресу 0100 (шестнадцатеричному). Каждый оператор перехода занимает 3 байта памяти.

ведет к реализации последовательности команд, управляющей стиранием ОЗУ.

Однако проведенное рассмотрение не раскрывает полной картины. Если осуществить трансляцию программы, то модуль рис. 9.10 представится в виде, показанном на рис. 9.11, где указаны абсолютные адреса каждого оператора перехода в таблице. Теперь использование указанных взаимосвязей так, как они приведены, порождает следующий порядок действий.

Предположим, что адрес первого оператора перехода в таблице принят в качестве базового. Тогда указатель нового адреса (NEWA) в таблице вычисляется как функция базового адреса и некоторой переменной V₁. В соответствии с рассмотренным выше соотношением имеем NEWA = Базовый адрес + (KWGT — 16), что вызывает определенное беспокойство, поскольку принятый базовый адрес, как показано на рис. 9.11, есть 0100₁₆. Как и прежде, KWGT = 19, и требуется определить адрес NEWA, соответствующий оператору JMP CLRAM. В соответст-

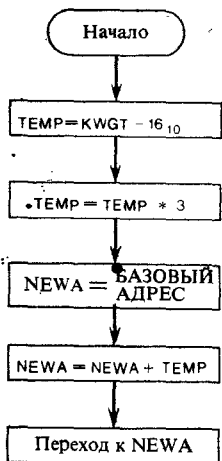


Рис. 9.12. Блок-схема вычисления точного адреса входа в таблицу операторов переходов.

7 0000	ORG 00	
8 0000 3A 00 10	LDA KWGT	АККУМУЛЯТОР=KWGT
9 0003 D6 10	SUI 16	АККУМУЛЯТОР=KWGT-16
10 0005 47	MOV B,A	ЗАПОМНИТЬ В РЕГИСТРЕ В
11 0006		
12 0006		
13 0006 87	ADD A	АККУМУЛЯТОР*2
14 0007 80	ADD B	АККУМУЛЯТОР*3
15 0008		
16 0008		
17 0008 01 00 00	LXI B,00	ОБНУЛЕНИЕ РЕГИСТРОВ В,С
18 000B 4F	MOV C,A	ЗАПОМНИТЬ В РЕГИСТРЕ С
19 000C 21 00 01	LXI H, TABLE	ЗАГРУЗИТЬ БАЗОВЫЙ АДРЕС ТАБЛИЦЫ
20 000F		
21 000F		
22 000F 09	DAD B	НОВЫЙ АДР.=БАЗ.АДР.+СОДЕРЖИМОЕ РЕГ. В
23 0010		
24 0010		
25 0010 E9	PCHL	ПЕРЕХОД ПО АДРЕСУ ТАБЛИЦЫ
26 0011		
27 0011		
28 0011		
29 0011		
30 0100	ORG 0100H	УСТАНОВИТЬ НАЧАЛО ТАБЛИЦЫ
31 0100	**** НАЧАЛО ТАБЛИЦЫ ****	
32 0100		
33 0100 C3 33 0F	TABLE JMP ENTER	
34 0103 C3 34 0F	JMP CENTY	
35 0106 C3 35 0F	JMP ADDSE	
36 0109 C3 36 0F	JMP CLRAM	
37 010C C3 37 0F	JMP PROG	
38 010F C3 38 0F	JMP VEY	
39 0112 C3 39 0F	JMP COPY	
40 0115 C3 3A 0F	JMP NA1	
41 0118 C3 3B 0F	JMP NA2	
42 011B		
43 011B	**** КОНЕЦ ТАБЛИЦЫ ****	

Рис. 9.13. Реализация блок-схемы рис. 9.12 в кодах символического языка микропроцессора 8080.

вии с рис. 9.11 этот адрес имеет значение 010В. И вышеприведенное соотношение дает

$$NEWA = \text{БАЗОВЫЙ АДРЕС} + (KWGT - 16),$$

$$NEWA = 0100 + (19_{10} - 16_{10}),$$

$$NEWA = 0100 + 0003,$$

$$NEWA = 0103.$$

Таким образом, получен адрес 0103 вместо требуемого 010В, указывающий оператор JMP CENTY (0103), а не требуемый оператор JMP CLRAM (010В). Ситуация ухудшается, когда $KWGT = 17_{10}$. В этом случае в результате аналогичных расчетов получается $NEWA = 0101_{16}$, т. е. абсолютный адрес, соответствующий среднему байту 3-байтовой команды, что является программной ошибкой. Анализируя положение, можно заметить, что в расчетах не учитывался тот факт, что каждая команда JMP занимает 3 байта памяти, а не 1 байт, как ранее предполагали. По этой причине вторая JMP команда смещена по отношению к первой не на один, а на три адреса. Для различных микропроцессоров трансляция команд JMP осуществляется по-разному. При этом необходима осторожность при вычислении адресов перехода по описанной схеме для заданного программного обеспечения.

Однако допущенная ошибка может быть легко исправлена путем изменения схемы расчета адреса перехода: $NEWA = \text{БАЗОВЫЙ АДРЕС} + (KWGT - 16) \times 3$.

С целью иллюстрации новой расчетной схемы рассмотрим прежний пример. Пусть значение $KWGT = 10_{10}$. Базовый адрес есть 0100₁₆, как это показано на рис. 9.11. Тогда значение нового адреса будет следующее:

$$NEWA = \text{БАЗОВЫЙ АДРЕС} + (KWGT - 16) \times 3,$$

$$NEWA = 0100_{16} + (19_{10} - 16_{10}) \times 3_{10},$$

$$NEWA = 0100_{16} + (3_{10} \times 3_{10}),$$

$$NEWA = 0109_{10}.$$

Из рис. 9.11 видно, что абсолютный адрес 0109 является правильным для входа в таблицу при $KWGT = 19$. На рис. 9.12 приведена блок-схема реализации описанных вычислений для микропроцессора 8085. Соответствующая программа представлена на рис. 9.13.

9.5. Программные средства реализации функции установки вдреса

Рассмотрев средства управления реализацией программных функций, перейдем к обсуждению программ, которые выполняют каждую из таких функций.

Отметим, что подпрограмма KEY INPUT может быть вызвана в любой точке программы. С помощью этой подпрограммы вызываются данные, необходимость в использовании которых возникает в данной точке.

Прежде всего рассмотрим функцию УСТАНОВКА АДРЕСА, которая может быть описана следующим образом. Функция установки адреса задает адрес памяти в интервале 0000—03FF₁₆, являющийся начальным адресом для проверки данных в ОЗУ₂ программ или для ввода данных. Если нажимается ключ функции установки адреса, система автоматически переходит в режим ожидания ввода трех шестнадцатеричных цифр. По завершении такой операции система использует эти цифры в качестве адреса ОЗУ₂.

Теперь на табло отображается адрес ОЗУ (0000—03FF) и данные, записанные в этой ячейке. Если необходимо скорректировать данные, вводятся две новые шестнадцатеричные цифры. Эти данные записываются в ОЗУ₂ при нажатии ключа ввода.

Если же мы хотим лишь проверить данные, то последовательным нажатием ключа ввода осуществляем перезапись исходных данных и увеличение на единицу адреса памяти. Блок-схема реализации подобной функции приведена на рис. 9.14.

Из рис. 9.14 видно, что реализация функции начинается с вывода на табло значения A30000₁₆ с целью проинформировать пользователя, что система ожидает ввода трех цифр адреса (ниблов). Затем программа их вводит. Отметим, что основной блок на схеме, помеченный как шаг 2, лишь устанавливает, что программа будет считывать три цифры. Считывание и отображение на табло цифр будет показано на примерах реальных программ. Описание же всех деталей основного блока шага 2 схемы может запутать общую картину.

Необходимые пояснения по программной реализации данного блока, позволяющие получить о нем детальное представление, будут даны при рассмотрении его описания с помощью мнемоники микропроцессора 8085. Приведенное предварительное упрощенное описание блок-схемы процесса имеет целью облегчить понимание основных идей. Поскольку для реализации программной функции необходимо ее подробное представление, то такое представление функции содержится в программных модулях, отображающих элементы блок-схемы, и может быть использовано при обсуждении материала.

Следующим шагом блок-схемы рис. 9.14 является чтение данных из ОЗУ₂ по адресу, который введен пользователем. Система должна рассчитать абсолютный адрес считывания, поскольку ячейке 0000 ОЗУ₂ в действительности соответствует адрес 1400₁₆. Этот системный адрес получается на основании таблицы распределения памяти рис. 9.1. Поэтому, если введен адрес 103₁₆, чтение будет осуществляться с абсолютно системного адреса $1400_{16} + 0103_{16} = 1503$. Отсюда становится ясно, что, до того как начать писать программы, необходимо составить хорошую таблицу распределения памяти.

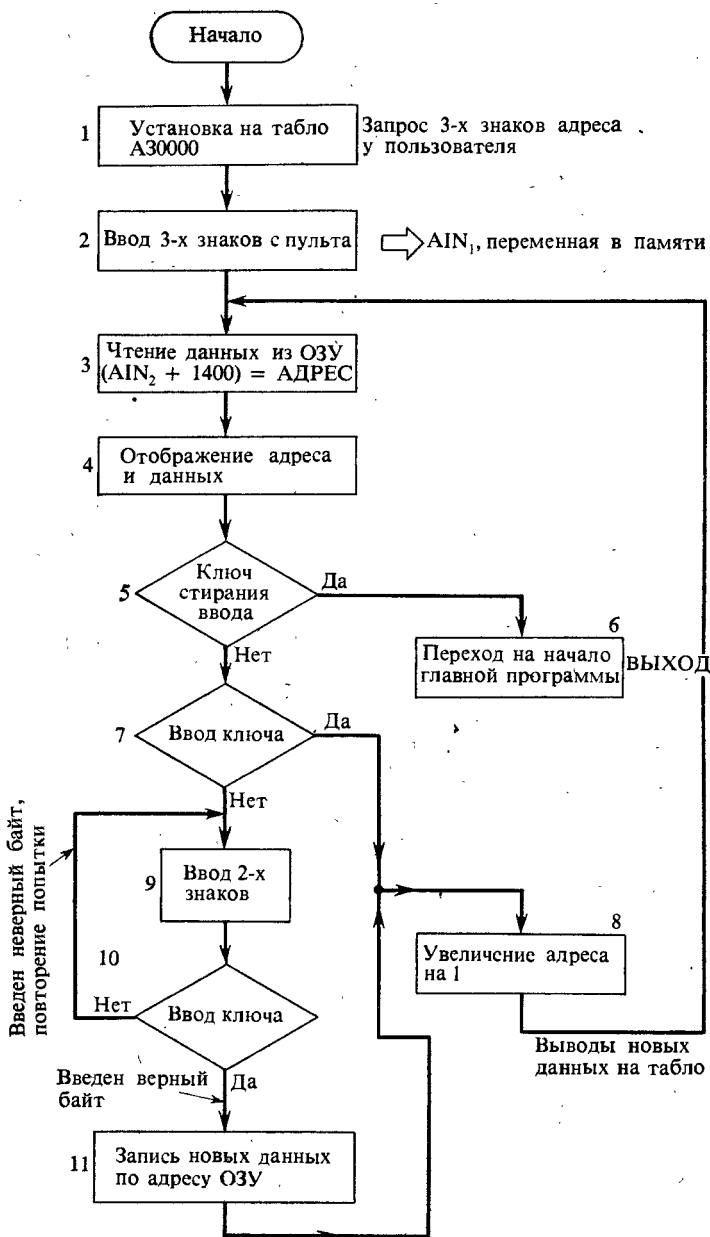


Рис. 9.14. Блок-схема функционирования системы при реализации функции установки адреса.

После чтения данных по заданному адресу ОЗУ₂ адрес и данные отображаются на табло. Это предусмотрено блоком 4 схемы рис. 9.14.

Далее система анализирует следующий ключ. Если таким ключом является ключ стирания ввода, реализация функции завершается и осуществляется возврат в главную управляющую программу. При нажатии же ключа ввода производится приращение адреса ОЗУ₂ на единицу (шаг 8) и отображение на табло нового адреса ОЗУ и данных, хранимых по этому адресу. Таким образом можно просмотреть память программ (ОЗУ₂) и проверить данные.

Если осуществляется ввод ключа данных 0—F (шаг 9), то программа рассматривает это как ввод новых данных по соответствующему адресу ОЗУ₂ и ожидает ввода двух цифр. После того как эти цифры введены, проверяется факт отработки ключа ввода (шаг 10), «подтверждающего» введенные данные. Если же вводится какой-либо иной ключ, выполнение програм-

```

1 0000      *
2 0000      *
3 0000      *
4 0000      * ПОДПРОГРАММА ADDSE.
5 0000      * ЭТА ПРОГРАММА ВВЕДЕТ АДРЕС ИЗ ТРЕХ ЦИФР С ПУЛЬТА,
6 0000      * ВЫВЕДЕТ АДРЕС НА ДИСПЛЕЙ И ЗАПОМНИТ ДАННЫЕ В
7 0000      * ПРОГРАММЕ ПЗУ ПО АДРЕСУ, УКАЗАННОМУ НА ДИСПЛЕЕ
8 0000      *
9 0000      *
10 0000      *
11 0000 3E A3  ADDSE  MVI  A,0A3H  УСТАНОВИТЬ АДРЕС ПЕРВОГО БАЙТА ПРОГ.
12 0002 D3 F2      OUT  OF2H
13 0004 AF      XRA  A
14 0005 D3 F1      OUT  OF1H
15 0007 D3 F0      OUT  OF0H  ВЫВЕСТИ НА ДИСПЛЕЙ A30000
16 0009      *
17 0009      *
18 0009 CD C3 00  CALL  KDET  ПРИНЯТЬ ПЕРВУЮ ЦИФРУ С ПУЛЬТА
19 000C 3A C2 00  LDA  KWGHT  АККУМУЛЯТОР=ПЕРВОЙ ЦИФРЕ
20 000F D3 F0      OUT  OF0H  ВЫВЕСТИ ПЕРВУЮ ЦИФРУ
21 0011 32 BF 00  STA  AIN1  ЗАПОМНИТЬ ПЕРВУЮ ЦИФРУ
22 0014      *
23 0014      *
24 0014 CD C3 00  CALL  KDET  ПРИНЯТЬ ВТОРУЮ ЦИФРУ С ПУЛЬТА
25 0017 3A C2 00  LDA  KWGHT  АККУМУЛЯТОР=ВТОРОЙ ЦИФРЕ
26 001A 32 BF 00  STA  AIN2  ЗАПОМНИТЬ ВТОРУЮ ЦИФРУ
27 001D 3A BE 00  LDA  AIN1  ПРИНЯТЬ ПЕРВУЮ ЦИФРУ ИЗ ПАМЯТИ
28 0020 17      RAL
29 0021 17      RAL
30 0022 17      RAL
31 0023 17      RAL
32 0024 E6 F0  AIN  OF0H  СДВИНУТЬ ВЛЕВО НА 4 БИТА
33 0026 32 C1 00  STA  AIN4  МАСКИРОВАТЬ МЛАДШИЕ 4 БИТА
34 0029 3A BF 00  LDA  AIN2  СОХРАНИТЬ ПРОМЕЖУТ. РЕЗ.
35 002C 21 C1 00  LXI  H,AIN4  АККУМУЛЯТОР=МЛАДШИЕ 4 БИТА ВТОРОЙ ЦИФ.
                          УСТАНОВИТЬ В H,L АДРЕС AIN4

```

Рис. 9.15. Реализация блок-схемы рис. 9.14 в кодах символического языка микропроцессора 8085.

36	002F	B6		ORA M	ЛОГ."ИЛИ"АККУМ. И ПЕРВОЙ ЦИФРЫ В AIN4
37	0030	D3	FO	OUT OFOH	ЗАПИСАТЬ НА ДИСПЛЕЙ XXXX--
38	0032		*		
39	0032		*		
40	0032	CD	C3 00	CALL KDET	ПРИНЯТЬ ТРЕТЬЮ ЦИФРУ
41	0035	3A	C2 00	LDA KWGHT	АККУМУЛЯТОР=ТРЕТЬЯ ЦИФРА
42	0038	32	C0 00	STA AIN3	ЗАПОМНИТЬ РЕЗУЛЬТАТ
43	003B	3A	BF 00	LDA AIN2	ВЫЗВАТЬ ВТОРУЮ ЦИФРУ
44	003E	17		RAL	
45	003F	17		RAL	
46	0040	17		RAL	
47	0041	17		RAL	СДВИГ ЦИФРЫ ВЛЕВО НА 4 БИТА
48	0042	E6	FO	AIN OFOH	ВЫДЕЛИТЬ 4 МЛАДШИХ БИТА
49	0044	21	C0 00	LXI H,AIN3	
50	0047	B6		ORA M	ЛОГ."ИЛИ"ВТОРОЙ И ТРЕТЬЕЙ ЦИФР
51	0048	D3	F1	OUT OFIH	ВЫВЕСТИ ЦИФРУ НА ДИСПЛЕЙ XX--XX
52	004A	32	BF 00	STA AIN2	ВТОРАЯ И ТРЕТЬЯ ЦИФРЫ В AIN2
53	004D		*		
54	004D		*		
55	004D	AF	.	XRA A	
56	004E	D3	FO	OUT OFOH	ДИСПЛЕЙ=XX--00
57	0050		*		
58	0050		*		
59	0050	3A	BE 00	LDA AIN1	
60	0053	D3	F2	OUT OF2H	ДИСПЛЕЙ=00 XXX00 ,ГДЕ XXX-АДРЕС
61	0055		*		
62	0055		*		
63	0055		*	ТЕПЕРЬ ЧИТАЕМ ДАННЫЕ ИЗ ПАМЯТИ	
64	0055		*	НИЖНЯЯ ГРАНИЦА=AIN2,ВЕРХНЯЯ ГРАНИЦА=AIN1	
65	0055		*		
66	0055	21	00 14	LXI H,1400H	H,L=НАЧАЛЬНЫЙ АДРЕС
67	0058	3A	BF 00	LDA AIN2	
68	005B	5F		MOV E,A	
69	005C	3A	BE 00	LDA AIN1	
70	005F	57		MOV D,A	D,E=АДРЕС ОЗУ
71	0060	D5		PUSH D	ЗАПИСАТЬ
72	0061	19		DAD D	H,L=НАЧАЛЬНЫЙ АДРЕС
73	0062		*		
74	0062		*		
75	0063	7E	NRFD	MOV A,M	ЗАГРУЗИТЬ В АККУМ. ДАННЫЕ ИЗ ОЗУ
76	0063	D3	FO	OUT OFOH	УСТАНОВИТЬ ДИСПЛЕЙ=0 XXXDD,ГДЕ DD-ДАНН
77	0065		*		
78	0065		*		
79	0065		*	МЫ ТОЛЬКО ЧТО ОКОНЧИЛИ ШАГ 4 БЛОК-СХЕМЫ	
80	0065		*		
81	0065		*		
82	0065	E5		PUSH H	ЗАПОМНИТЬ АДРЕС
83	0066	CD	C3 00	CALL KDET	ЖДАТЬ СЛЕДУЮЩЕГО НАЖАТИЯ КЛАВИШИ
84	0069	3A	C2 00	LDA KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
85	006C	FE	11	CPI 17	
86	006F	CA	B9 00	JZ ADEX	ДА,ИДТИ НА ВЫХОД ИЗ ЭТОЙ ПРОГРАММЫ
87	0071	FE	10	CPI 16	НАЖАТА КЛАВИША ENTER??
88	0073	CA	AB 00	JZ STPB	ДА,ИДТИ НА ШАГ 8 БЛОК-СХЕМЫ
89	0076		*		
90	0076		*		
91	0076		*	НЕОБХОДИМО ПРИНЯТЬ ДВЕ НОВЫЕ ЦИФРЫ И ЗАПОМНИТЬ ИХ В ОЗУ	
92	0076		*		
93	0076	CD	C3 00	CALL KDET	ПРИНЯТЬ НОВУЮ ЦИФРУ
94	0079	3A	C2 00	LDA KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
95	007C	32	C1 00	STA AIN4	ПРОМЕЖУТОЧНЫЕ ЗАПОМИНАНИЯ
96	007F	D3	FO	OUT OFOH	ЗАПИСЬ НА ДИСПЛЕЙ
97	0081	CD	C3 00	CALL KDET	ПРИНЯТЬ СЛЕДУЮЩУЮ ЦИФРУ
98	0084	3A	C2 00	LDA KWGHT	
99	0087		*		

100	0087	32	00 00	STA	AIN3	ВРЕМЕННО СОХРАНИТЬ В ПАМЯТИ
101	008A	3A	C1 00	LDA	AIN4	ВЫЗВАТЬ ПЕРВУЮ ЦИФРУ
102	008D	17		RAL		
103	008E	17		RAL		
104	008F	17		RAL		
105	0090	17		RAL		СДВИНУТЬ ВЛЕВО НА 4 БИТА
106	0091	21	00 00	LXI	H,AIN3	
107	0094	B6		ORA	M	ЛОГ. "ИЛИ" ПЕРВОЙ И ВТОРОЙ ЦИФР
108	0095	32	00 00	STA	AIN3	СОХРАНИТЬ ДАННЫЕ В ПАМЯТИ
109	0098	D3	F0	OUT	OF0H	ВЫВЕСТИ НА ДИСПЛЕЙ
110	009A			*		
111	009A			*		
112	009A			*	ПРОБЕРИТЬ ДАННЫЕ	
113	009A			*		
114	009A	CD	C3 00	CALL	KDET	ПРИНЯТЬ КЛАВИШУ
115	009D	3A	C2 00	LDA	KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
116	00A0	FE	10	CPI	16	КЛАВИША ENTER???
117	00A2	C2	76 00	JNZ	TNIB	НЕТ,ПОВТОРНО ЗАПРОСИТЬ НОВЫЕ ДАННЫЕ
118	00A5			*		
119	00A5			*		
120	00A5			*	ЗАПИСАТЬ ДАННЫЕ В ПАМЯТЬ	
121	00A5			*		
122	00A5	3A	00 00	LDA	AIN3	ВЫЗВАТЬ ДАННЫЕ
123	00A8	E1		POP	H	ВОССТАНОВИТЬ АДРЕС В ПАМЯТИ
124	00A9	77		MOV	M,A	ДАННЫЕ В ПАМЯТЬ
125	00AA	E5		PUSH	H	
126	00AB			*		
127	00AB			*		
128	00AB	E1		STPB	POP H	
129	00AC	23		INX	H	УВЕЛИЧИТЬ АДРЕС НА 1
130	00AD	D1		POP	D	ВОССТАНОВИТЬ АДРЕС ОЗУ
131	00AE	13		INX	D	
132	00AF	D5		PUSH	D	СОХРАНИТЬ ЗНАЧЕНИЕ
133	00B0	7B		MOV	A,E	
134	00B1	D3	F1	OUT	OF1H	ВЫВЕСТИ АДРЕС НА ДИСПЛЕЙ
135	00B3	7A		MOV	A,D	
136	00B4	D3	F2	OUT	OF2H	ВЫВЕСТИ АДРЕС НА ДИСПЛЕЙ
137	00B6	C3	62 00	JMP	NRED	ЕЩЕ РАЗ НАЧАТЬ ОПЕРАЦИЮ
138	00B9			*		
139	00B9			*		
140	00B9	E1		ADEX	POP H	
141	00BA	D1		POP	D	
142	00BB	C3	C4 00	JMP	BEGIN	ИДТИ НА НАЧАЛО УПРАВЛЯЮЩЕЙ ПРОГРАМ.
143	00BE			*		
144	00BE			*		
145	00BE			*		
146	00BE			*		
147	00BE			*	НЕОБХОДИМО УКАЗАТЬ МАКЕТ РАЗМЕЩЕНИЯ ПЕРЕМЕННЫХ	
148	00BE			*		
149	00BE	00		AIN1	NOP	
150	00BF	00		AIN2	NOP	
151	00C0	00		AIN3	NOP	
152	00C1	00		AIN3	NOP	
153	00C2	00		KWGHT	NOP	
154	00C3	00		KDET	NOP	
155	00C4	00		BEGIN	NOP	
156	00C5			END		ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА

0 ERRORS

12 SYMBOLS

мы не будет продолжено, чему соответствует условие «нет» шага 10.

Если обрабатывается ключ ввода, то данные, считанные на шаге 9, записываются в ОЗУ₂, как это показано на шаге 11. После этого осуществляется переход к шагу 8, на котором адрес ОЗУ₂ увеличивается на 1. По завершении шага 8 снова производится переход к шагу 3. Программа реализации функции УСТАНОВКИ АДРЕСА приведена на рис. 9.15.

9.6. Программные средства реализации функции СТИРАНИЯ ОЗУ

Перейдем к рассмотрению программных средств, предназначенных для реализации системной функции СТИРАНИЯ ОЗУ, для чего прежде всего определим эту функцию. Функция СТИРАНИЕ ОЗУ заключается в записи единиц (FF₁₆) во все ячейки ОЗУ₂.

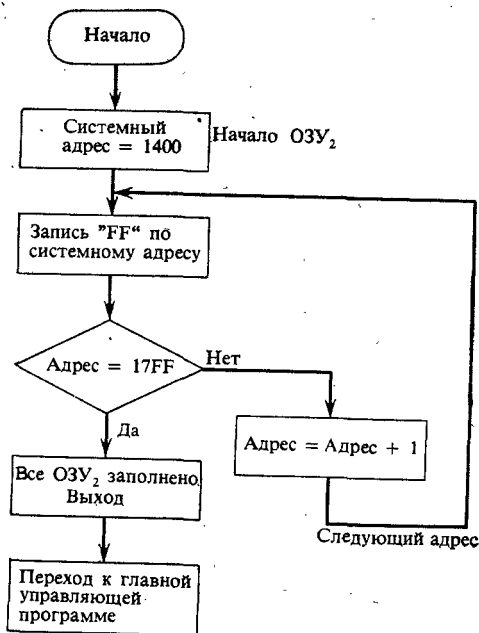
Укрупненная блок-схема реализации рассматриваемой функции приведена на рис. 9.16. После записи единиц во все ячейки ОЗУ₂ осуществляются возврат в основную программу и ожидание ввода новой функции. Программная реализация блок-схемы рис. 9.16 средствами символического языка микропроцессора 8085 показана на рис. 9.17.

9.7. Программные средства реализации функции программирования

Программирование ППЗУ осуществляется в том случае, когда возникает необходимость хранения информации (отлаженных программ), длительное время используемой без перепрограммирования. Подобные запоминающие устройства, сохраняющие информацию при отключении питания, находят применение во многих областях. ППЗУ обладают еще одним важным свойством, которое заключается в том, что при необходимости записанную информацию можно стереть, а устройство — перепрограммировать в соответствии с новыми потребностями. При программировании ППЗУ требуемая программа вначале размещается в другом ЗУ, в нашем случае это ОЗУ₂, а затем пересылается при выполнении операции программирования в ППЗУ, где и хранится до тех пор, пока не возникнет потребность ее затереть.

Рассмотрим теперь программные средства, необходимые для программирования ППЗУ. Система устанавливает ППЗУ в режим программирования. Данные из ОЗУ₂ поступают на выходные линии ППЗУ. Далее на вывод программирования ППЗУ подается импульс. Выдача импульса программирования производится последовательно для каждого адреса (0000—03FF₁₆).

Рис. 9.16. Блок-схема работы системы при реализации функции стирания ОЗУ.



```

1 0000 *****
2 0000 *
3 0000 * ПОДПРОГРАММА ЧИСТКИ ОЗУ
4 0000 *
5 0000 *
6 0000 *****
7 0000 ORG 0000 УСТАНОВИТЬ НАЧ. АДРЕС ПРОГРАММЫ
8 0000 3E FF CLRAM MVI A,0FFH АККУМУЛЯТОР=FF
9 0002 21 00 14 LXI H,1400H В H,L НАЧАЛЬНЫЙ АДРЕС ОЗУ
10 0005 77 CL1 MOV M,A FF В ПАМЯТЬ
11 0006 F5 PUSH PSW СОХРАНИТЬ СОДЕРЖАНИЕ АККУМУЛЯТОРА
12 0007 3E 17 MVI A,17H
13 0009 BC CMP H ПРОВЕРИТЬ СТАРШИЙ БАЙТ=ПРЕДЕЛ. ЗНАЧ.
14 000A CA 12 00 JZ LCHK ДА,ИДТИ НА ПРОВЕРКУ МЛАДШЕГО БАЙТА
15 000D F1 POP PSW ВОССТАНОВИТЬ АККУМУЛЯТОР
16 000E 23 INX H УВЕЛИЧИТЬ АДРЕС ОЗУ
17 000F C3 05 00 JMP CL1 ОБРАБОТАТЬ СЛЕДУЮЩИЙ АДРЕС
18 0012 F1 POP PSW ВОССТАНОВИТЬ АККУМУЛЯТОР
19 0013 BD CMP L ПРОВЕРИТЬ МЛАДШИЙ БАЙТ=FF
20 0014 CA 1B 00 JZ BEGIN КОНЕЦ ЭТОЙ ПОДПРОГРАММЫ
21 0017 23 INX H УВЕЛИЧИТЬ АДРЕС ОЗУ
22 0018 C3 05 00 JMP CL1 ПОВТОРИТЬ ЗАНОВО С ДРУГИМ АДРЕСОМ
23 001B *
24 001B *
25 001B *
26 001B * НЕОБХОДИМ АДРЕС ДЛЯ МЕТКИ BEGIN
27 001B *
28 001B 00 BEGIN NOP ЭТО ПУСТОЙ ОПЕРАТОР АССЕМБЛЕРА
29 001C *
30 001C END ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА
  
```

0 ERRORS 4 SYMBOLS

Рис. 9.17. Реализация блок-схемы рис. 9.16 в кодах символического языка микропроцессора 8085.

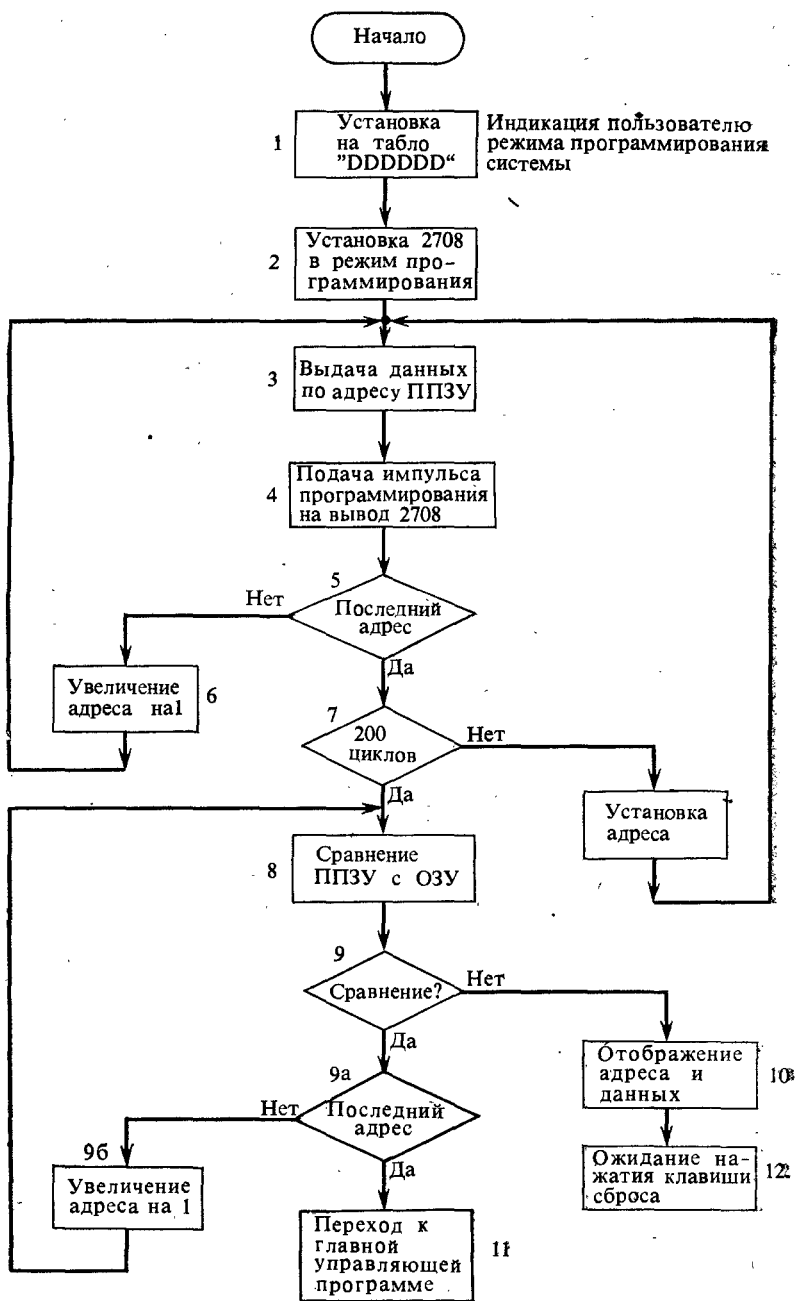


Рис. 9.18. Блок-схема работы системы в режиме программирования.

```

1 0000
2 0000
3 0000
4 0000
5 0000
6 0000
7 0000
8 0000
9 0000
10 0000
11 0000
12 0000
13 0000 3E DD
14 0002 D3 F0
15 0004 D3 F1
16 0006 D3 F2
17 0008 3E 00
18 000A D3 32
19 000C 32 03 14
20 000F 3C
21 0010 D3 34
22 0012
23 0012 11 00 00
24 0015 21 00 14
25 0018
26 0018 7E
27 0019 D3 30
28 001B 7B
29 001C D3 33
30 001E 7A
31 001F D3 34
32 0021
33 0021
34 0021
35 0021 06 08
36 0023 3E 03
37 0025 D3 35
38 0027 05
39 0028 C2 27 00
40 002B 3E 01
41 002D D3 35
42 002F
43 002F
44 002F
45 002F 7B
46 0030 FE FF
47 0032 C2 3B 00
48 0035 7A
49 0036 FE 03
50 0038 CA 40 00
51 003B 23
52 003C 13
53 003D C3 18 00
54 0040
55 0040
56 0040
57 0040 3A 03 14
58 0043 3C
59 0044 32 03 14
60 0047 FE C8

```

```

*****
*
* РЕАЛИЗАЦИЯ КОМАНДЫ PROG ДЛЯ 8080.
* ЭТА ПРОГРАММА СЛУЖИТ ДЛЯ УПРАВЛЕНИЯ ПЕРЕПРОГ. ПЗУ.
* ИНФОРМАЦИЯ ОЗУ ХРАНИТСЯ НАЧИНАЯ С ЯЧЕЙКИ 1400H.
* АДРЕСА ПЕРЕПРОГРАММИРУЕМОГО ПЗУ НАЧИНАЮТСЯ С 0000.
*
***** ПРОГРАММИСТ ДЖИМ КОФФРОН *****
*
ORG 0000 УСТАНОВИТЬ НАЧ. АДРЕС ДЛЯ АССЕМБЛЕРА
*
PROG MVI A,0DDH
OUT 0F0H
OUT 0F1H
OUT 0F2H ВЫВЕСТИ НА ДИСПЛЕЙ DDDDDD
MVI A,00
OUT 32H РАЗРЕШИТЬ ПЕРЕДАЧУ ДАННЫХ В ПРОГ. ПЗУ
STA CNT1 ОБНУЛИТЬ СЧЕТЧИК ЦИКЛОВ
INR A
OUT 34H CS/WE=12 VOLTS
*
PROG3 LXI D,0000H ЗАГРУЗИТЬ НАЧ. АДРЕС ПРОГ. ПЗУ
LXI H,1400H ЗАГРУЗИТЬ НАЧ. АДРЕС ОЗУ
*
PROG2 MOV A,M ДАННЫЕ ИЗ ОЗУ В АККУМУЛЯТОР
OUT 3H ВЫВЕСТИ ДАННЫЕ В ПРОГ. ПЗУ
MOV A,F АО-A7 В АККУМУЛЯТОР
OUT 33H АО-A7 В ПРОГ. ПЗУ
MOV A,D А8-A9 В АККУМУЛЯТОР
OUT 34H А8-A9 В ПРОГ. ПЗУ
*
* ПРИМЕНИТЬ ПРОГРАММИРУЮЩИЙ ИМПУЛЬС
*
MVI B,08
MVI A,03
OUT 35H ВКЛЮЧИТЬ CS/WE И ПРОГ. ИМПУЛЬС
PULSE DCR B
JNZ PULSE ШИРИНА ИМПУЛЬСА
MVI A,01
OUT 35H ВЫКЛЮЧИТЬ ПРОГ. ИМПУЛЬС
*
* ПОСЛЕДНИЙ АДРЕС ПРОГРАММИРОВАЛСЯ?
*
MOV A,E
CPI 0FFH ПОСЛЕДНИЙ БАЙТ АО-A7
JNZ PROG1
MOV A,D
CPI 03 ПОСЛЕДНИЙ БАЙТ А8-A9
JZ LOPC ПОСЛЕДНИЙ АДРЕС (ДА)
PROG1 INX H
INX D УВЕЛИЧИТЬ АДРЕСА ОЗУ И ПРОГ. ПЗУ
JMP PROG2 ОБРАБОТАТЬ СЛЕДУЮЩИЙ АДРЕС
*
* ПОСЛЕДНИЙ РАЗ ОБРАБАТЫВАЛИСЬ ВСЕ 1024 АДРЕСА?
*
LOPC LDA CNT1 ЗАГРУЗИТЬ СЧЕТЧИК
INR A
STA CNT1
CPI 200 200 ЦИКЛОВ???

```

Рис. 9.19. Реализация блок-схемы рис. 9.18 в кодах символического языка микропроцессора 8085.

```

61 0049 C2 12 00      JNZ PROG3      НЕ ЗОЩИКЛОВ,ПОВТОРНЫЙ СТАРТ
62 004C      *
63 004C      * ПРОГРАММИРОВАНИЕ ОКОНЧЕНО,ТЕПЕРЬ ПРОВЕРИТЬ
64 004C      *
65 004C 3E 01      MVI A,01      ЗАПРЕТИТЬ ПЕРЕДАЧУ ДАННЫХ В ПРОГ. ПЗУ
66 004E D3 32      OUT 32H
67 0050 3D      DCR A      АККУМУЛЯТОР=0
68 0051 D3 35      OUT 35H      CS/WE = 0.0 V
69 0053      *
70 0053      * ИНИЦИАЛИЗИРОВАТЬ АДРЕС ПРОГ. ПЗУ И ОЗУ
71 0053      *
72 0053 11 00 00      LXI D,00
73 0056 21 78 05      LXI H,1400      АДРЕС ОЗУ
74 0059      *
75 0059 7B      PROG4 MOV A,E      АО-А7 В АККУМУЛЯТОР
76 005A D3 33      OUT 33H      АДРЕС В ПРОГ. ПЗУ
77 015C 7A      MOV A,D      А8-А9 В АККУМУЛЯТОР
78 005D D3 34      OUT 34H      А8-А9 В ПРОГ. ПЗУ
79 005F DB 31      IN 31H      ЧИТАТЬ ДАННЫЕ ИЗ ПРОГ.ПЗУ В АККУМУЛ.
80 0061 EE      CMP M      СРАВНИТЬ ДАННЫЕ В ОЗУ И В ПРОГ. ПЗУ
81 0062 C2 76 00      JNZ PROG5      НЕ РАВНЫ,ОШИБКА
82 0065      *
83 0065      *
84 0065      *
85 0065      * АДРЕС ПОСЛЕДНИЙ?
86 0065      *
87 0065 7B      MOV A,E
88 0066 FE FF      CPI 0FFH      ПОСЛЕДНИЙ АО-А7
89 0068 C2 71 00      JNZ PROG5      НЕ ПОСЛЕДНИЙ
90 006B 7A      MOV A,D      ПОСЛЕДНИЙ А8-А9
91 006C FE 03      CPI 03
92 006E CA 02 14      JZ BEGIN      ПОСЛЕДНИЙ.НАЧАТЬ НОВУЮ КОМАНДУ
93 0071      *
94 0071      * УВЕЛИЧИТЬ АДРЕСА ОЗУ И ПРОГ. ПЗУ
95 0071      *
96 0071 23      PROG5 INX H
97 0072 13      INX D
98 0073 C3 59 00      JMP PROG4      СРАВНЕНИЕ ПО СЛЕДУЮЩИМ АДРЕСАМ
99 0076      *
100 0076      * БЛОК ОБРАБОТКИ ОШИБОК
101 0076      *
102 0076 7B      PROG6 MOV A,E      АО-А7 В АККУМУЛЯТОР
103 0077 D3 F0      OUT 0F0H
104 0079 7A      MOV A,D      А8-А9 В АККУМУЛЯТОР
105 007A D3 F1      OUT 0F1H      НА ДИСПЛЕЙ
106 007C DB 31      IN 31H      ПРИНЯТЬ ДАННЫЕ ИЗ ПРОГ.ПЗУ
107 007E D3 F2      OUT 0F2H      НА ДИСПЛЕЙ
108 0080      *
109 0080      * ЗАЩИТИТЬ ПРОГ. ПОКА НЕ БУДЕТ НАЖАТА КЛАВИША СЕ
110 0080      *
111 0080 CD 00 14      PROG6 CALL KDET
112 0083 3A 01 14      LDA KWGT
113 0086 FE 16      CPI 16H      НАЖАТА КЛАВИША СЕ???
114 0088 C2 80 00      JNZ PROG6
115 008B C3 02 14      JMP BEGIN      ВЕРНУТЬСЯ И ОБРАБОТАТЬ ДРУГУЮ КОМАН.
116 008E      *
117 008E      * УКАЗАТЬ РАЗМЕЩЕНИЕ ПЕРЕМЕННЫХ
118 008E      *
119 008E      *
120 008E      *

```

```

121 008E      *   УСТАНОВИТЬ ПЕРЕМЕННЫЕ.
122 008E      *
123 008E      *****
124 008E      *
125 008E      KDET   EQU   1400H
126 008E      KWGT   EQU   1401H
127 008E      BEGIN  EQU   1402H
128 008E      CNT1   EQU   1403H
129 008E      *
130 008E      *
131 008E      END                                END ДЛЯ АССЕМБЛ
    
```

0 ERRORS

14 SYMBOLS

Через каждые 200 шагов программирования в диапазоне адресов 0000—03FF система выполняет операцию сравнения данных, записанных в ППЗУ с исходными данными, хранящимися в ОЗУ₂. В случае когда вся информация, записанная в ППЗУ, совпадает с исходной информацией из ОЗУ₂, считается, что операция программирования была выполнена успешно. Если же данные для какого-либо адреса ППЗУ не совпадают с соответствующими данными ОЗУ₂, то этот адрес и данные отображаются на табло. Блок-схема описанной процедуры приведена на рис. 9.18.

Из рис. 9.18 видно, что работа подпрограммы начинается с записи на табло числа DD_{DDDD}₁₆. Это делается с целью индикации факта реализации операции программирования ППЗУ. Для того чтобы полностью запрограммировать ППЗУ, необходимо затратить более минуты, так что индикация этой операции необходима.

Далее система переводит ППЗУ в режим программирования (шаг 2), что осуществляется, как известно, путем подачи напряжения +12 В на вывод CS/WE устройства 2708. Затем реализуется циклическое повторение шагов 3, 4, 5, 6, реализующих собственно программирование устройства 2708 с помощью данных из ОЗУ₂.

По завершении полного прохода программирования всех 1024 ячеек осуществляется проверка на выполнение 200 циклов программирования, что показано на рис. 9.18 (шаг 7). Если 200 циклов не выполнено, производится переход к шагу 3.

В противном случае осуществляется верификация запрограммированных данных, что выражается шагами 8 и 9.

Несравнение данных означает наличие ошибок при программировании устройства. В этом случае на табло отображается а) адрес первой по порядку ячейки, где обнаружено несравнение и б) данные в ППЗУ, заключенные по этому адресу, что выражается шагами 10 и 12. При полном сравнении данных осуществляется индикация правильности программирования путем выдачи на табло значения FDDDDD и производится пере-

дача управления в начало головной программы, что показано на рис. 9.18 (шаг 11).

На рис. 9.19 приведена программная реализация блок-схемы рис. 9.18 средствами символического языка микропроцессора 8085.

9.8. Программные средства реализации функции верификации

Перейдем к рассмотрению функции верификации, заключающейся в сравнении данных ППЗУ с данными из ОЗУ₂. В слу-

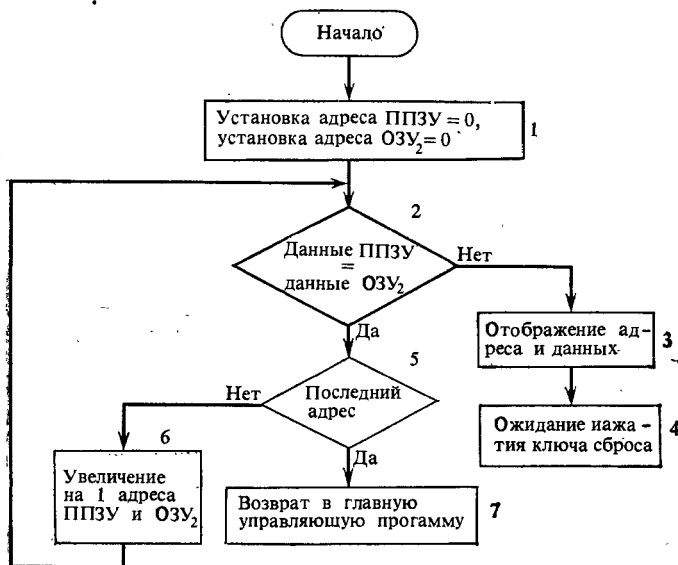


Рис. 9.20. Блок-схема работы системы при реализации функции верификации.

чае несовпадения какого-либо элемента данных на табло отображается адрес этого элемента и соответствующие данные. Рассматриваемая функция особенно необходима в том случае, когда существует сомнение в правильном программировании ППЗУ. Она используется также для правильности затирания информации в ППЗУ. С этой целью вначале производится стирание ОЗУ₂ и затем — проверка правильности стирания ППЗУ с помощью ключа верификации.

Блок-схема реализации функции верификации приведена на рис. 9.20, по которой видно, что вначале система устанавливает значение 0000 для адреса сравнения (шаг 1). Далее осущест-

```

1 0000 *****
2 0000 *
3 0000 * НА РИС. 9.21 ПОКАЗАНА ПРОГРАММА VFY, БЛОК-СХЕМА
4 0000 * КОТОРОЙ ИЗОБРАЖЕНА НА РИС. 9.20.
5 0000 *ПРОГРАММИСТ ДЖИМ КОФФОН
6 0000 *****
7 0000 *
8 0000
9 0000 ORG 0000 УКАЗАТЬ АССЕМБЛЕРУ НАЧАЛО ПРОГ.
10 0000 *
11 0000 * УСТАНОВИТЬ РЕЖИМ ЧТЕНИЯ ИЗ ПРОГ.ПЗУ
12 0000 *
13 0000 AF VFY XRA A
14 0001 3C INR A
15 0002 D3 32 OUT 32H ОБЕСПЕЧИТЬ РЕЖИМ ЧТЕНИЯ
16 0004 AF XRA A
17 0005 D3 35 OUT 35H CS=0.0V, PGM=0.0V
18 0007 D3 33 OUT 33H АДРЕС А0-А7=0
19 0009 D3 34 OUT 34H АДРЕС А8-А9=0
20 000B 11 00 00 LXI D,0000 УСТАНОВИТЬ АДРЕС ПРОГ.ПЗУ
21 000E 21 00 14 LXI H,1400H УСТАНОВИТЬ АДРЕС ОЗУ
22 0011 *
23 0011 * НАЧАТЬ ЦИКЛ ЧТЕНИЯ И ПРОВЕРКИ
24 0011 *
25 0011 DB 31 VFY1 IN 31H ЧИТАТЬ ДАННЫЕ ИЗ ПРОГ. ПЗУ
26 0013 BE CMP M СРАВНИТЬ ДАННЫЕ ОЗУ И ПРОГ. ПЗУ
27 0014 C2 31 00 JNZ VFAIL ЕСЛИ НЕ РАВНЫ, ТО НА VFAIL
28 0017 3E 13 MVI A,13H
29 0019 BA CMP D ПРОВЕРИТЬ СТАРШИЙ БАЙТ, ПОСЛЕД. АДРЕС
30 001A C2 26 00 JNZ VNEХ НЕ ПОСЛЕДНИЙ, ПРИНЯТЬ СЛЕДУЮЩИЙ АДРЕС
31 001D 3E FF MVI A,OFFH
32 001F BB CMP E ПРОВЕРИТЬ МЛАДШИЙ БАЙТ, ПОСЛЕДНИЙ АДРЕС
33 0020 C2 26 00 JNZ VNEХ НЕ ПОСЛЕДНИЙ, ПРИНЯТЬ СЛЕДУЮЩИЙ АДРЕС
34 0023 C3 3C 00 JMP START НАЗАД К ОСНОВНОЙ ПРОГРАММЕ
35 0026 13 VNEХ INX D
36 0027 7A MOV A,D НОВЫЙ АДРЕС ПРОГ. ПЗУ
37 0028 D3 34 OUT 34H ВЫДАТЬ ДАННЫЕ
38 002A 7B MOV A,E
39 002B D3 33 OUT 33H ВЫДАТЬ ДАННЫЕ
40 002D 23 INX H НОВЫЙ АДРЕС ОЗУ
41 002E C3 11 00 JMP VFY1 СРАВНИТЬ СЛЕДУЮЩИЕ ДАННЫЕ
42 0031 D3 F0 VFAIL OUT OFOH ВЫДАТЬ НА ДИСПЛЕЙ----ХХ
43 0033 7B MOV A,E
44 0034 D3 F1 OUT OF1H
45 0036 7A MOV A,D
46 0037 D3 F2 OUT OF2H ВЫВОД АДРЕСА НА ДИСПЛЕЙ
47 0039 C3 39 00 VOUT JMP VOUT ДАТЬ ПЕРЕРЫВАНИЯ
48 003C *
49 003C *
50 003C ***** ТЕПЕРЬ ВВЕДЕМ ФИКТИВНУЮ ПЕРЕМЕННУЮ *****
51 003C *
52 003C 00 START NOP
53 003D END END ДЛЯ АССЕМБЛЕРА

0 ERRORS 6 SYMBOLS

```

Рис. 9.21. Реализация блок-схемы рис. 9.20 в кодах символического языка микропроцессора 8085.

вляется сравнение данных по адресу 0 из ОЗУ₂ с данными по адресу 0 ППЗУ. Если эти данные не сравниваются, на табло выводятся адрес данных и собственно данные, что показано на схеме (шаги 3 и 4). При сравнении данных производится увеличение на 1 адреса (шаг 6).

При последовательном увеличении адреса осуществляется сравнение данных по каждому адресу. В случае когда данные

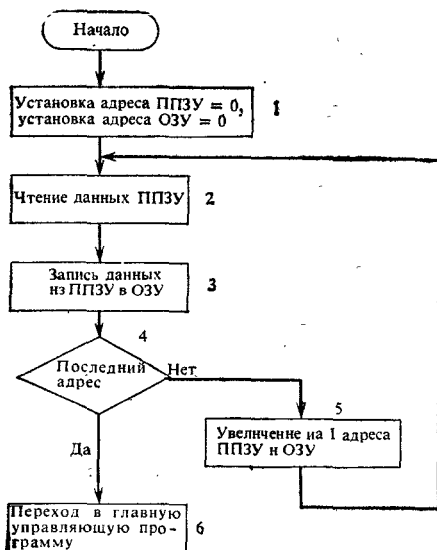


Рис. 9.22. Блок-схема работы системы при реализации функции копирования.

```

1 0000 *****
2 0000 *
3 0000 * НА РИС.9.23 ПОКАЗАНА ПРОГРАММА COPY,БЛОК-СХЕМА
4 0000 * КОТОРОЙ ИЗОБРАЖЕНА НА РИС.9.22
5 0000 *
6 0000 * ПРОГРАММИСТ ДЖИМ КОФФРОН
7 0000 *
8 0000 *****
9 0000 *
10 0000 ORG 000H
11 0000 *
12 0000 * РЕЖИМ ЧТЕНИЯ ИНФОРМАЦИИ ИЗ ПРОГ. ПЗУ
13 0000 *
14 0000 AF COPY XRA A
15 0001 3C INR A АККУМУЛЯТОР=1
16 0002 D3 32 OUT 32H
17 0004 AF XRA A
18 0005 D3 35 OUT 35H CS=0.0V,PGM=0.0V
19 0007 11 00 00 LXI D,0000 АДРЕС ПРОГ. ПЗУ=0
20 000A 21 00 14 LXI H,1400H УСТАНОВИТЬ АДРЕС ОЗУ
21 000D 7B COPY1 MOV A,E
22 000E D3 33 OUT 33H АДРЕС A0-A7
23 0010 7A MOV A,D
24 0011 D3 34 OUT 34H АДРЕС A8-A9
25 0013 DB 31 IN 31H ЧИТАТЬ ИНФОРМАЦИЮ ИЗ ПРОГ. ПЗУ
26 0015 77 MOV M,A ВЫВЕСТИ ИНФОРМАЦИЮ В ПЗУ
27 0016 3E 13 MVI A,13H
28 0018 BA CMP D ПРОВЕРКА НА ПОСЛЕДНИЙ АДРЕС
29 0019 C2 22 00 JNZ CNEХ ЕСЛИ АДРЕС НЕ ПОСЛЕДНИЙ,ТО НА CNEХ
30 001C 3E FF MVI A,0FFH
31 001F BB CMP B МЛАДШИЙ БАЙТ ПОСЛЕДНЕГО АДРЕСА
32 001F CA 27 00 JZ START ЕСЛИ АДРЕС ПОСЛЕДНИЙ,ТО КОНЕЦ
33 0022 23 CNEХ INX H СЛЕДУЮЩИЙ АДРЕС ОЗУ
34 0023 13 INX D СЛЕДУЮЩИЙ АДРЕС ПЗУ
35 0024 C3 0D 00 JMP COPY1 ПРИНЯТЬ СЛЕДУЮЩИЕ ДАННЫЕ
36 0027 *
37 0027 * ТЕПЕРЬ ВВЕДЕМ ФИКТИВНУЮ ПЕРЕМЕННУЮ
38 0027 *
39 0027 00 START NOP
40 0028 END ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА

```

0 ERRORS

4 SYMBOLS

```

1 0000 *****
2 0000 *
3 0000 *   УПРАВЛЕНИЕ ПРОГРАММИРОВАНИЕМ ПРОГ. ПЗУ
4 0000 *
5 0000 *   ***** ПРОГРАММИСТ ДИМ КОФРОН *****
6 0000 *
7 0000 *
8 0000   ORG 00   УСТАНОВИТЬ НАЧАЛЬНЫЙ АДРЕС
9 0000 *
10 0000 *   ЗАДАТЬ НАЧАЛЬНЫЕ ЗНАЧЕНИЯ ДЛЯ ВСЕХ ПЕРЕМЕННЫХ
11 0000 *
12 0000 31 FF 13   LXI SP,13FFH   НАЧ. ЗНАЧЕНИЕ УКАЗАТЕЛЯ СТЕКА
13 0003 AF   BEGIN XRA A   НАЧ. ЗНАЧЕНИЯ ФЛАГОВ, ОБНУЛИТЬ АККУМ.
14 0004 FB   EI   РАЗРЕШИТЬ ПРЕРЫВАНИЯ
15 0005 D3 F0   OUT OF0H   НОЛЬ НА ДИСПЛЕЙ
16 0007 D3 F1   OUT OF1H
17 0009 D3 F2   OUT OF2H   ВЫДАТЬ НА ДИСПЛЕЙ 000000
18 000B 32 00 10   STA KTIME   KTIME=0
19 000E 32 01 10   STA CFLAG   CFLAG=0
20 0011 32 02 10   STA KWGHT   KWGHT=0
21 0014 32 03 10   STA KCOMP   KCOMP=0
22 0017 3C   INR A   АККУМУЛЯТОР=1
23 0018 32 04 10   STA KROW   KROW=00000001
24 001B *
25 001B *
26 001B *   УСТАНОВИТЬ АДРЕСА ПЕРЕМЕННЫХ
27 001B *
28 001B   KTIME EQU 1000H
29 001B   CFLAG EQU KTIME+1
30 001B   KWGHT EQU CFLAG+1
31 001B   KCOMP EQU KWGHT+1
32 001B   KROW EQU KCOMP+1
33 001B   NROW EQU KROW+1
34 001B   CNT1 EQU NROW+1
35 001B   AIN1 EQU CNT1+1
36 001B   AIN2 EQU AIN1+1
37 001B   AIN3 EQU AIN2+1
38 001B   AIN4 EQU AIN3+1
39 001B *
40 001B *   НАЧАЛО ОСНОВНОЙ ПРОГ. ОБРАБОТКИ ВВОДА КОМАНД
41 001B *
42 001B CD 64 00   CALL KDET   ЖДАТЬ НАЖАТИЯ КЛАВИШИ
43 001E *
44 001E 3A 02 10   LDA KWGHT   ВЫЧИСЛИТЬ ЗНАЧЕНИЕ КЛАВИШИ
45 0021 FE 16   CPI 16H   КЛАВИША ВВОДА КОМАНДЫ???
46 0023 CA 2C 00   JZ BEGN1
47 0026 DA 2C 00   JC BEGN1   ЕСЛИ ПЕРЕХОДЫ СРАБОТАЛИ, ТО КОМ. ПРАВ.
48 0029 C3 03 00   JMP BEGN1   ИГНОРИРОВАТЬ ВВОД
49 002C *
50 002C *   ВЫЧИСЛИТЬ АДРЕС ПЕРЕХОДА
51 002C *   РИС. 9.13
52 002C *
53 002C 3A 02 10   BEGN1 LDA KWGHT
54 002F D6 10   SUI 16   АККУМУЛЯТОР=KWGHT-16
55 0031 47   MOV B,A
56 0032 87   ADD A   АККУМУЛЯТОР*2
57 0033 80   ADD B   АККУМУЛЯТОР*3
58 0034 *
59 0034 01 00 00   LXI B,00

```

Рис. 9.24. Полная программа для микропроцессора 8085, реализующая все операции системы.

60	0037	4F	MOV	C,A	ВЫЧИСЛЕННОЕ СМЕЩЕНИЕ В РЕГИСТР C
61	0038	21 3D 00	LXI	H, TABLE	ЗАГРУЗИТЬ БАЗОВЫЙ АДРЕС ТАБЛИЦЫ
62	003B	09	DAD	B	ПРИБАВИТЬ СМЕЩЕНИЕ К БАЗОВОМУ АДРЕСУ
63	003C	E9	PNHL		ИДТИ К ВЫЧИСЛЕННОМУ АДРЕСУ ТАБЛИЦЫ
64	003D		*		
65	003D		*		
66	003D		*	ТАБЛИЦА ДЛЯ АДРЕСОВ ПЕРЕХОДОВ	
67	003D		*		
68	003D	C3 58 00	TABLE JMP	ENTER	
69	0040	C3 5B 00	JMP	CENTRY	
70	0043	C3 44 01	JMP	ADDSE	
71	0046	C3 02 02	JMP	CIRAM	
72	0049	C3 1D 02	JMP	PROG	
73	004C	C3 AB 02	JMP	VXY	
74	004F	C3 F2 02	JMP	COPY	
75	0052	C3 5E 00	JMP	NA1	
76	0055	C3 61 00	JMP	NA2	
77	0058		*		
78	0058		***	КОНЕЦ ТАБЛИЦЫ ENTRYC	*****
79	0058		*		
80	0058		*	КОМАНДЫ ENTRY, CENTRY, NA1, NA2	
81	0058		*		
82	0058	C3 03 00	ENTER JMP	BEGIN	
83	005B	C3 03 00	CENTRY JMP	BEGIN	
84	005E	C3 03 00	NA1 JMP	BEGIN	
85	0061	C3 03 00	NA2 JMP	BEGIN	
86	0064		*		
87	0064		*		
88	0064		*****		
89	0064		*		
90	0064		*	НАЧАЛО РИС. 4.20	
91	0064		*		
92	0064	CD 2F 01	KDET CALL	OROW	ВЫВЕСТИ АКТИВНЫЙ НАБОР
93	0067	CD 1F 01	CALL	COLM	ВЫЗВАТЬ ПОДПРОГРАММУ COLM
94	006A	3A 01 10	LDA	CFLAG	НЕОБХОДИМО ПРОВЕРИТЬ ФЛАЖОК СТОЛБЦА
95	006D	FE 00	CPI	00H	ПРОВЕРИТЬ АКТИВНОСТЬ СТОЛБЦА
96	006F	CA 64 00	JZ	KDET	НЕ АКТИВНЫЙ СТОЛБЕЦ, ПОВТОРИТЬ ЦИКЛ
97	0072	CD F5 00	CALL	KEYW	АКТИВНЫЙ СТОЛБЕЦ, КАКОЙ ИМЕННО???
98	0075		*		
99	0075		*	КОНЕЦ РИС. 4.20	
100	0075		*		
101	0075		*	НАЧАЛО РИС. 4.21	
102	0075		*		
103	0075		*****	БЫЛИ НАЖАТЫ КЛАВИШИ	*****
104	0075		*		
105	0075		*****	(ШАГ 1)	
106	0075		*		
107	0075	3A 00 10	LDA	KTIME	ПЕРЕЗАГРУЗИТЬ KTIME ИЗ ПАМЯТИ
108	0078		*		
109	0078		*		
110	0078		*****	(ШАГ 2)	
111	0078		*		
112	0078	FE 00	CPI	00H	KTIME=0??
113	007A	CA 9F 00	JZ	KCOL1	ДА, ЭТО ПЕРВЫЙ РАЗ
114	007D		*		
115	007D		***	НЕ ПЕРВЫЙ РАЗ	*****
116	006D		*		
117	007D		*****	(ШАГ 7)	
118	007D		*		
119	007D	3A 02 10	LDA	KWGH1	КАКАЯ ЦИФРА БЫЛА НАВРАНА???

120	0080	4F		MOV	C, A	РЕГИСТР C=KWGHT
121	0081	3A 03 10		LDA	KCOMP	РЕГИСТР A=KCOMP
122	0084	B9		CMF	C	KCOMP=KWGHT??
123	0085	CA 8F 00		JZ	KCLO2	ДА, ОНИ РАВНЫ!!!!!!
124	0088		*			
125	0088		***** (ШАГ 8)			
126	0088		*			
127	0088	AF		XRA	A	ОНИ НЕ РАВНЫ
128	0089	32 00 10		STA	KTIME	ПЕРЕЗАПИСАТЬ KTIME
129	008C	C3 AA 00		JMP	KCLO3	ИДТИ НА ПЕРЕЗАГРУЗКУ АКТИВ. НАВОР
130	008F		*			
131	008F		***** (ШАГ 9)			
132	008F		*			
133	008F	3A 00 10	KCLO2	LDA	KTIME	ВЫЗВАТЬ KTIME ИЗ ПАМЯТИ
134	0092	3C		INR	A	KTIME=KTIME+1
135	0093	32 00 10		STA	KTIME	KTIME=KTIME+1
136	0096		*			
137	0096		***** (ШАГ 10)			
138	0096		*			
139	0096	FE 32		CPI	50	KTIME=50??
140	0098	C2 AA 00		JNZ	KCLO3	ЕЩЕ НЕ 50
141	009B	CD B2 00		CALL	KOPN	ПРОВЕРИТЬ ГОТОВНОСТЬ ПУЛЬТА
142	009E	C9		RET		КОНЕЦ ЭТОЙ ПОДПРОГРАММЫ
143	009F		*			
144	009F		***** (ШАГ 3)			
145	009F		*			
146	009F	3A 02 10	KCLO1	LDA	KWGHT	
147	00A2	32 03 10		STA	KCOMP	KCOMP=KWGHT
148	00A5		*			
149	00A5		***** (ШАГ 4)			
150	00A5		*			
151	00A5	3E 01		MVI	A, 01	
152	00A7	32 00 10		STA	KTIME	УСТАНОВИТЬ KTIME=1
153	00AA		*			
154	00AA		***** (ШАГ 5)			
155	00AA		*			
156	00AA	3E 01	KCLO3	MVI	A, 01	
157	00AC	32 04 10		STA	KROW	УСТАНОВИТЬ АКТИВНЫЙ НАВОР=00000001
158	00AF		*			
159	00AF		***** (ШАГ 6)			
160	00AF		*			
161	00AF	C3 64 00		JMP	KDET	ИДТИ К НАЧАЛУ ПРОГРАММЫ
162	00B2		*			
163	00B2		*			
164	00B2		***** НАЧАЛО ПОДПРОГРАММЫ *****			
165	00B2		*			
166	00B2		***** НАЧАЛО ПОДПРОГРАММЫ *****			
167	00B2		*			
168	00B2		***** НАЧАЛО ПОДПРОГРАММЫ *****			
169	00B2		*			
170	00B2		*			
171	00B2		РИС.4.22			
172	00B2		ПРОГРАММА ПОДГОТОВКИ ПУЛЬТА			
173	00B2		*			
174	00B2		***** (ШАГ 1)			
175	00B2		*			
176	00B2		*			
177	00B2		*			
178	00B2	3E 01	KOPN	MVI	A, 01	
179	00B4	32 04 10		STA	KROW	KROW=00000001

```

180 OOB7 3D          DCR A
181 OOB8 32 05 10   STA NROW      NROW=00000000
182 OOB8           *
183 OOB8           ***** (ШАГ 2)
184 OOB8           *
185 OOB8 32 00 10   STA KTIME      KTIME=0
186 OOB8           *
187 OOB8           ***** (ШАГ 3)
188 OOB8           *
189 OOB8 CD 2F 01   KOPN1 CALL OROW  ВЫВЕСТИ НАБОР
190 OOC1           *
191 OOC1           ***** (ШАГ 4)
192 OOC1           *
193 OOC1 CD 1 F 01   CALL COLM      ВЫВЕСТИ ДАННЫЕ СТОЛБЦА
194 OOC4           *
195 OOC4           ***** (ШАГ 5)
196 OOC4           *
197 OOC4 3A 01 10   LDA CFLAG      ВВЕСТИ CFLAG
198 OOC7 FE 00       CPI 00          CFLAG=0??
199 OOC9 C2 B2 00   JNZ KOPN       ПУЛЬТ ЕЩЕ НЕ СВОБОДЕН
200 OOC          *
201 OOC          ***** (ШАГ 9)
202 OOC          *
203 OOC 3A 05 10   LDA NROW        ВВЕСТИ АКТИВНЫЙ НАБОР
204 OOCF 3C         INR A
205 OOD0 32 05 10   STA NROW      ЗАПИСАТЬ АКТИВНЫЙ НАБОР
206 OOD3           *
207 OOD3           ***** (ШАГ 10)
208 OOD3           *
209 OOD3 FE 05       CPI 05
210 OOD5 C2 BE 00   JNZ KOPN1      СКАНИРОВАНИЕ НЕ ЗАКОНЧЕНО
211 OOD8           *
212 OOD8           ***** (ШАГ 6)
213 OOD8           *
214 OOD8 3A 00 10   LDA KTIME
215 OODB 3C         INR A
216 OODC 32 00 10   STA KTIME
217 OODF 47         MOV B,A        РЕГИСТР В=KTIME
218 OOE0           *
219 OOE0           ***** (ШАГ 6A)
220 OOE0           *
221 OOE0 AF         XRA A
222 OOE1 32 05 10   STA NROW      ЗНАЧЕНИЕ НАБОРА=0
223 OOE4           *
224 OOE4           ***** (ШАГ 7)
225 OOE4           *
226 OOE4 3E 32       MVI A,50
227 OOE6 B8         CMP B          KTIME=50???
228 OOE7 C2 BE 00   JNZ KOPN1      НЕТ, СКАНИРОВАНИЕ ЕЩЕ РАЗ
229 OOE8           *
230 OOE8           ***** (ШАГ 8)
231 OOE8           *
232 OOE8 C9         RET            ВЫХОД ИЗ ПОДПРОГРАММЫ ПОДГОТОВКИ
233 OOE8           *
234 OOE8           *
235 OOE8           *****
236 OOE8           *
237 OOE8           * ПОДПРОГРАММА KOUT
238 OOE8           *
239 OOE8           * ЭТА ПОДПРОГРАММА ВЫВЕДЕТ ДАННЫЕ С ПУЛЬТА НА ДИСПЛЕЙ

```

```

240 00EB *
241 00EB *****
242 00EB *
243 00EB 3A 02 10 KOUT LDA KWGHT ДАННЫЕ С ПУЛЬТА В АККУМУЛЯТОРЕ
244 00EE D3 F0 OUT OF0H
245 00F0 D3 F1 OUT OF1H
246 00F2 D3 F2 OUT OF2H
247 00F4 C9 RET
248 00F5 *
249 00F5 *****
250 00F5 *
251 00F5 * ПОДПРОГРАММА KEYW
252 00F5 *
253 00F5 * ЭТА ПОДПРОГРАММА УСТАНОВИТ ЗНАЧЕНИЕ НАЖАТОЙ КЛАВИШИ
254 00F5 *
255 00F5 *****
256 00F5 *
257 00F5 ***** БЛОК-СХЕМА ЭТОЙ ПРОГРАММЫ НА РИС.4.16
258 00F5 *
259 00F5 ***** (ШАГ 1)
260 00F5 *
261 00F5 AF KEYW XRA A ОБНУЛИТЬ ФЛАЖКИ И АККУМУЛЯТОР
262 00F6 3A 01 10 LDA CFLAG ПРИНЯТЬ ФЛАЖОК СТОЛБЦА ИЗ ПАМЯТИ
263 00F9 *
264 00F9 ***** (ШАГ 2)
265 00F9 *
266 00F9 06 00 MVI B,00 ОБНУЛИТЬ СЧЕТЧИК
267 00FB *
268 00FB ***** (ШАГ 3)
269 00FB *
270 00FB 1F KEYW1 RAR СДВИНУТЬ ВПРАВО НА 1 БИТ, 0 В D7
271 00FC FE 00 CPI 00
272 00FE *
273 00FE ***** (ШАГ 4)
274 00FE *
275 00FE CA 05 01 JZ KEYW2 РЕГИСТР В=0,1,2,3,4 (ШАГ 6)
276 0101 *
277 0101 ***** (ШАГ 5)
278 0101 *
279 0101 04 INR B
280 0102 C3 FB 00 JMP KEYW1 УВЕЛИЧИТЬ ЗНАЧ.СЧЕТ. И ИДТИ НА ПОВТОР
281 0105 *
282 0105 ***** (ШАГ 7)
283 0105 *
284 0105 AF KEYW2 XRA A ОБНУЛИТЬ ФЛАЖКИ
285 0106 3A 04 10 LDA KROW ВЫЗВАТЬ АКТИВНЫЙ НАБОР
286 0109 *
287 0109 ***** (ШАГ 8)
288 0109 *
289 0109 0E 00 MVI C,00 ОБНУЛИТЬ СЧЕТЧИК T2
290 010B *
291 010B ***** (ШАГ 9)
292 010B *
293 010B 1F KEYW3 RAR СДВИГ ВПРАВО НА 1 БИТ, 0 В D7
294 010C FE 00 CPI 00
295 010E *
296 010E ***** (ШАГ 10)
297 010E *
298 010E CA 15 01 JZ KEYW4 ЕСЛИ НОЛЬ, ТО ВЫПОЛНЕНО
299 0111 *

```

```

300 0111 ***** (ШАГ 14)
301 0111 *
302 0111 OC INR C УВЕЛИЧИТЬ ЗНАЧЕНИЕ СЧЕТЧИКА
303 0112 C3 0B 01 JMP KEYW3 ИДТИ НА ПОВТОР
304 0115 *
305 0115 ***** (ШАГ 11)
306 0115 *
307 0115 AF KEYW4 XRA A ОВНУЛИТЬ ФЛАЖКИ
308 0116 78 MOV A,B АККУМУЛЯТОР=СЧЕТЧИК СТОЛБЦОВ(T1)
309 0117 17 RAL B*2
310 0118 17 RAL B*4
311 0119 8D ADD B B*5
312 011A 81 ADD C (B*5)+C=0 - 24
313 011B *
314 011B ***** (ШАГ 12)
315 011B *
316 011B 32 02 10 STA KWGHT
317 011E *
318 011E ***** (ШАГ 13)
319 011E *
320 011E C9 RET ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
321 011F *
322 011F *****
323 011F *
324 011F ПОДПРОГРАММА COLM
325 011F *
326 011F * ЭТА ПОДПРОГРАММА ОПРЕДЕЛИТ АКТИВНЫЙ СТОЛБЕЦ,
327 011F * ЕСЛИ ОН СУЩЕСТВУЕТ.
328 011F *
329 011F *****
330 011F *
331 011F * ЭТА ПОДПРОГРАММА-РИС.4.15
332 011F *
333 011F *****
334 011F *
335 011F ***** (ШАГ 1)
336 011F *
337 011F AF COLM XRA A
338 0120 32 01 10 STA CFLAG УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА=0
339 0123 *
340 0123 ***** (ШАГ 2)
341 0123 *
342 0123 DB FE IN OFEH ВХОДНЫЕ ДАННЫЕ ИЗ ПОРТА FE
343 0125 *
344 0125 ***** (ШАГ 3)
345 0125 *
346 0125 F6 ED ORI OEON УСТАНОВИТЬ БИТЫ D5,D6,D7=1
347 0127 *
348 0127 ***** (ШАГ 4)
349 0127 *
350 0127 FE FF CPI OFFH
351 0129 C8 RZ ШАГ(4A)
352 012A *
353 012A ***** (ШАГ 5)
354 012A *
355 012A 2F CMA ИНЕРТИРОВАТЬ ДАННЫЕ
356 012B *
357 012B ***** (ШАГ 6)
358 012B *
359 012B 32 01 10 STA CFLAG УСТАНОВИТЬ ФЛАЖОК СТОЛБЦА

```

```

360 012E C9          RET          ВЫЙТИ ИЗ ЭТОЙ ПОДПРОГРАММЫ
361 012F          *
362 012F          *
363 012F          *
364 012F          * ПОДПРОГРАММА ВЫВОДА АКТИВНОГО НАВОРА НА ПУЛЬТ
365 012F          *
366 012F          * ИМЯ ЭТОЙ ПОДПРОГРАММЫ - OROW
367 012F          *
368 012F          * ЭТО РИС.4.12
369 012F          *
370 012F          *
371 012F          *
372 012F          *
373 012F          * РИС.4.12          ПОДПРОГ. К РИС.4.10 В МНЕМОНИКЕ 8088
374 012F          *
375 012F          *
376 012F          *
377 012F          *
378 012F 3A 04 10 OROW LDA KROW      ПОВТОРНЫЙ ВЪЗОВ АКТИВНОГО НАВОРА
379 0132          *
380 0132          * ШАГ 2
381 0132          *
382 0132 FE 10          CPI 10H      СКАНИРОВАНИЕ НАВОРА ЗАКОНЧЕНО???
383 0134 C2 3C 01      JNZ OROW1    ЕСЛИ НЕТ,ТО ШАГ 2B
384 0137          *
385 0137          * ШАГ 2A
386 0137          *
387 0137 3E 01          MVI A,01H    ПЕРЕМЕСТИТЬ АКТИВНЫЙ НАВОР
388 0139 C3 3D 01      JMP ST3      ИДТИ НА ШАГ 3
389 013C          *
390 013C          * ШАГ 2B
391 013C          *
392 013C 07          OROW1 RLC      СДВИНУТЬ ДАННЫЕ ВЛЕВО, О В ДО
393 013D          *
394 013D          * ШАГ 3
395 013D          *
396 013D 32 04 10 ST3 STA KROW      СОХРАНИТЬ АКТИВНЫЙ НАВОР
397 0140          *
398 0140          * ШАГ 4
399 0140          *
400 0140 2F          CMA          ИНВЕРТИРОВАТЬ СЛОВО НАВОРА
401 0141          *
402 0141          * ШАГ 5
403 0141          *
404 0141 D3 FE          OUT FEH      ВЫВЕСТИ АКТИВНЫЙ НАВОР В ПОРТ FE
405 0143          *
406 0143          * ШАГ 6
407 0143          *
408 0143 C9          RET
409 0144          *
410 0144          *
411 0144          *
412 0144          *
413 0144          *
414 0144          * ПОДПРОГРАММА ADDSE.РИС.9.15
415 0144          * ЭТА ПРОГРАММА ВВЕДЕТ АДРЕС ИЗ ТРЕХ ЦИФР С ПУЛЬТА,
416 0144          * ВЫВЕДЕТ АДРЕС НА ДИСПЛЕЙ И ЗАПОМНИТ ДАННЫЕ В
417 0144          * ПРОГРАММЕ ПЗУ ПО АДРЕСУ,УКАЗАННОМУ НА ДИСПЛЕЕ
418 0144          *
419 0144          *

```

420	0144	*			
421	0144	TE A3	ADDSE	MVI	A,0A3H
422	0146	D3 F2		OUT	OF2H
423	0148	AF		XRA	A
424	0149	D3 F1		OUT	OF1H
425	014B	D3 FO		OUT	OF0H
426	014D	*			ВЫВЕСТИ НА ДИСПЛЕЙ A30000
427	014D	*			
428	014D	CD 64 00		CALL	KDET
429	0150	3A 02 10		LDA	KWGHT
430	0153	D3 FO		OUT	OF0H
431	0155	32 07 10		STA	AIN1
432	0158	*			ПРИНЯТЬ ПЕРВУЮ ЦИФРУ С ПУЛЬТА
433	0158	*			АККУМУЛЯТОР=ПЕРВОЙ ЦИФРЕ
434	0158	CD 64 00		CALL	KDET
435	015B	3A 02 10		LDA	KWGHT
436	015E	32 08 10		STA	AIN2
437	0161	3A 07 10		LDA	AIN1
438	0164	17		RAL	
439	0165	17		RAL	
440	0166	17		RAL	
441	0167	17		RAL	СДЕВИНУТЬ ВЛЕВО НА 4 БИТА
442	0168	E6 FO		AIN	OF0H
443	016A	32 0A 10		STA	AIN4
444	016D	3A 08 10		LDA	AIN2
445	0170	21 0A 10		LXI	H,AIN4
446	0173	B6		ORA	M
447	0174	D3 FO		OUT	OF0H
448	0176	*			ЛОГ."ИЛИ"АККУМ. И ПЕРВОЙ ЦИФ.В AIN4
449	0176	*			ЗАПИСАТЬ НА ДИСПЛЕЙ XXXX--
450	0176	CD 64 00		CALL	KDET
451	0179	3A 02 10		LDA	KWGHT
452	017C	32 09 10		STA	AIN3
453	017F	3A 08 10		LDA	AIN2
454	0182	17		RAL	
455	0183	17		RAL	
456	0184	17		RAL	
457	0185	17		RAL	СДЕВИГ ЦИФРЫ ВЛЕВО НА 4 БИТА
458	0186	E6 FO		AIN	OF0H
459	0188	21 09 10		LXI	H,AIN3
460	018B	B6		ORA	M
461	018C	D3 F1		OUT	OF1H
462	018E	32 08 10		STA	AIN2
463	0191	*			ВЫВЕСТИ ЦИФРУ НА ДИСПЛЕЙ
464	0191	*			ВТОРАЯ И ТРЕТЬЯ ЦИФРЫ В AIN2
465	0191	AF		XRA	A
466	0192	D3 FO		OUT	OF0H
467	0194	*			ДИСПЛЕЙ=XX--00
468	0194	*			
469	0194	3A 07 10		LDA	AIN1
470	0197	D3 F2		OUT	OF2H
471	0199	*			ДИСПЛЕЙ=0 XXX00 ;ГДЕ XXX-АДРЕС
472	0199	*			
473	0199	*			ТЕПЕРЬ ЧИТАЕМ ДАННЫЕ ИЗ ПАМЯТИ
474	0199	*			НИЖНЯЯ ГРАНИЦА=AIN2,ВЕРХНЯЯ ГРАНИЦА=AIN1
475	0199	*			
476	0199	21 00 14		LXI	H,1400H
477	019C	3A 08 10		LDA	AIN2
478	019F	5F		MOV	E,A
479	01A0	3A 07 10		LDA	AIN1

480	01A3	57	MOV	D,A	D,E=АДРЕС ОЗУ
481	01A4	D5	PUSH	D	ЗАПИСАТЬ
482	01A5	19	DAD	D	H,L=НАЧАЛЬНЫЙ АДРЕС
483	01A6				
484	01A6				
485	01A6	7E	NRED	MOV A,M	ЗАГРУЗИТЬ В АККУМ. ДАННЫЕ ИЗ ОЗУ
486	01A7	D3 FO		OUT OFOH	УСТАНОВИТЬ ДИСП.=0XXXDD,ГДЕ DD-ДАНН.
487	01A9				
488	01A9				
489	01A9				
490	01A9				
491	01A9				
492	01A9	E5			
493	01AA	CD 64 00	PUSH	H	ЗАПОМНИТЬ АДРЕС
494	01AD	3A 02 10	CALL	KDET	ЖДАТЬ СЛЕДУЮЩЕГО НАЖАТИЯ КЛАВИШИ
495	01B0	FE 11	LDA	KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
496	01B2	CA FD 01	CPI	17	НАЖАТА КЛАВИША CLEAR??
497	01B5	FE FE 10	JZ	ADEX	ДА,ИДТИ НА ВЫХОД ИЗ ЭТОЙ ПРОГРАММЫ
498	01B7	CA FF 01	CPI	16	НАЖАТА КЛАВИША ENTER??
499	01BA		JZ	STR8	ДА,ИДТИ НА ШАГ 8 БЛОК-СХЕМЫ
500	01BA				
501	01BA				
502	01BA				
503	01BA	CD 64 00	TNIR	CALL KDET	ПРИНЯТЬ НОВУЮ ЦИФРУ
504	01B0	3A 02 10	LDA	KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
505	01CD	32 0A 10	STA	AIN4	ПРОМЕЖУТОЧНЫЕ ЗАПОМИНАНИЯ
506	01C3	D3 FO	OUT	OFOH	ЗАПИСЬ НА ДИСПЛЕЙ
507	01C5	CD 64 00	CALL	KDET	ПРИНЯТЬ СЛЕДУЮЩУЮ ЦИФРУ
508	01C8	3A 02 10	LDA	KWGHT	
509	01CB				
510	01CB	32 09 10	STA	AIN3	ВРЕМЕННО СОХРАНИТЬ В ПАМЯТИ
511	01CE	3A 0A 10	LDA	AIN4	ВЫЗВАТЬ ПЕРВУЮ ЦИФРУ
512	01D1	17	RAL		
513	01D2	17	RAL		
514	01D3	17	RAL		
515	01D4	17	RAL		СДВИНУТЬ ВЛЕВО НА 4 БИТА
516	01D5	21 09 10	LXI	H,AIN3	
517	01D8	B6	ORA	M	ЛОГ."ИЛИ"ПЕРВОЙ И ВТОРОЙ ЦИФР
518	01D9	32 09 10	STA	AIN3	СОХРАНИТЬ ДАННЫЕ В ПАМЯТИ
519	01DC	D3 FO	OUT	OFOH	ВЫВЕСТИ НА ДИСПЛЕЙ
520	01DE				
521	01DE				
522	01DE				
523	01DE				
524	01DE	CD 64 00	CALL	KDET	ПРИНЯТЬ КЛАВИШУ
525	01E1	3A 02 10	LDA	KWGHT	АККУМУЛЯТОР=ЗНАЧЕНИЕ КЛАВИШИ
526	01E4	FE 10	CPI	16	КЛАВИША ENTER???
527	01E6	C2 BA 01	JNZ	TNIB	НЕТ,ПОВТОРНО ЗАПРОСИТЬ НОВЫЕ ДАННЫЕ
528	01E9				
529	01E9				
530	01E9				
531	01E9				
532	01E9	3A 09 10	LDA	AIN3	ВЫЗВАТЬ ДАННЫЕ
533	01EC	E1	POP	M	ВОССТАНОВИТЬ АДРЕС В ПАМЯТИ
534	01ED	77	MOV	H,A	ДАННЫЕ В ПАМЯТЬ
535	01EE	E5	PUSH	H	
536	01FF				
537	01EF				
538	01EF	E1	STR8	POP H	
539	01FO	23		INX H	УВЕЛИЧИТЬ АДРЕС НА 1


```

540 01F1 D1      POP D      ВОССТАНОВИТЬ АДРЕС ОЗУ
541 01F2 13      INX D
542 01F3 D5      PUSH D     СОХРАНИТЬ ЗНАЧЕНИЕ
543 01F4 7B      MOV A,E
544 01F5 D3 F1    OUT OF1H   ВЫВЕСТИ АДРЕС НА ДИСПЛЕЙ
545 01F7 7A      MOV A,D
546 01F8 D3 F2    OUT OF2H   ВЫВЕСТИ АДРЕС НА ДИСПЛЕЙ
547 01FA C3 A6 01 JMP NRED   ЕЩЕ РАЗ НАЧАТЬ ОПЕРАЦИЮ
548 01FD
549 01FD
550 01FD E1      ADEX      POP H
551 01FE D1      POP D
552 01FF C3 03    JMP BEGIN   ИДТИ НА НАЧАЛО УПРАВЛЯЮЩЕЙ ПРОГРАМ.
553 0202
554 0202
555 0202
556 0202
557 0202
558 0202
559 0202
560 0202
561 0202
562 0202
563 0202 3E FF   CLRAM     MVI A,OFFH  АККУМУЛЯТОР=FF
564 0204 21 00 14 LXI H,1400H  В Н.Л. НАЧАЛЬНЫЙ АДРЕС ОЗУ
565 0207 77      CL1       MOV M,A    FF В ПАМЯТЬ
566 0208 F5      PUSH PSW     СОХРАНИТЬ СОДЕР. АККУМУЛЯТОРА
567 0209 3E 17   MVI A,17H
568 020B BC      CMP H       ПРОВЕРИТЬ СТАРШИЙ БАЙТ=ПРЕДЕЛ. ЗНАЧ.
569 020C CA 14 02 JZ LCHK     ДА, ИДТИ НА ПРОВЕРКУ МЛАДШЕГО БАЙТА
570 020F F1      POP PSW     ВОССТАНОВИТЬ АККУМУЛЯТОР
571 0210 23      INX H       УВЕЛИЧИТЬ АДРЕС ОЗУ
572 0211 C3 07 02 JMP CL1     ОБРАБОТАТЬ СЛЕДУЮЩИЙ АДРЕС
573 0214 F1      LCHK      POP PSW     ВОССТАНОВИТЬ АККУМУЛЯТОР
574 0215 BD      CMP L       ПРОВЕРИТЬ МЛАДШИЙ БАЙТ=FF
575 0216 CA 03 00 JZ BEGIN   КОНЕЦ ЭТОЙ ПОДПРОГРАММЫ
576 0219 23      INX H       УВЕЛИЧИТЬ АДРЕС ОЗУ
577 021A C3 07 02 JMP CL1     ПОВТОРИТЬ ЗАНОВО С ДРУГИМ АДРЕСОМ
578 021D
579 021D
580 021D
581 021D
582 021D
583 021D
584 021D
585 021D
586 021D
587 021D
588 021D
589 021D
590 021D
591 021D
592 021D
593 021D 3E DD   PROG     MVI A,DDH
594 021F D3 FO    OUT OF0H
595 0221 D3 F1    OUT OF1H
596 0223 D3 F2    OUT OF2H   ВЫВЕСТИ НА ДИСПЛЕЙ DDDDDD
597 0225 3E 00    MVI A,00
598 0227 D3 32    OUT 32H     РАЗРЕШИТЬ ПЕРЕДАЧУ ДАННЫХ В ПРОГ.ПЗУ
599 0229 32 06 10 STA CNT1   ОБНУЛИТЬ СЧЕТЧИК ЦИКЛОВ

```

600	022C	3C		INR	A	
601	022D	D3 34		OUT	34H	CS/WE = 12 VOLTS
602	022F		*			
603	022F	11 00 00	PROG3	LXI	D,0000H	ЗАГРУЗИТЬ НАЧ.АДРЕС ПРОГ. ПЗУ
604	0232	21 00 14		LXI	H,1400H	ЗАГРУЗИТЬ НАЧ. АДРЕС ОЗУ
605	0235		*			
606	0235	7E	PROG2	MOV	A,M	ДАННЫЕ ИЗ ОЗУ В АККУМУЛЯТОР
607	0236	D3 30		OUT	30H	ВЫВЕСТИ ДАННЫЕ В ПРОГ. ПЗУ
608	0238	7B		MOV	A,E	А0-А7 В АККУМУЛЯТОР
609	0239	D3 33		OUT	33H	А0-А7 В ПРОГ. ПЗУ
610	023B	7A		MOV	A,D	А8-А9 В АККУМУЛЯТОР
611	023C	D3 34		OUT	34H	А8-А9 В ПРОГ. ПЗУ
612	023E		*			
613	023E		*			ПРИМЕНИТЬ ПРОГРАММИРУЮЩИЙ ИМПУЛЬС
614	023E		*			
615	023E	06 08		MVI	B,08	
616	0240	3E 03		MVI	A,03	
617	0242	D3 35		OUT	35H	ВКЛЮЧИТЬ CS/WE И ПРОГ. ИМПУЛЬС
618	0244	05	PULSE	DCR	B	
619	0245	C2 44 02		JNZ	PULSE	ШИРИНА ИМПУЛЬСА
620	0248	3E 01		MVI	A,01	
621	024A	D3 35		OUT	35H	ВЫКЛЮЧИТЬ ПРОГ. ИМПУЛЬС
622	024C		*			
623	024C		*			ПОСЛЕДНИЙ АДРЕС ПРОГРАММИРОВАЛСЯ?
624	024C		*			
625	024C	7B		MOV	A,E	
626	024D	FE FF		CPI	0FFH	ПОСЛЕДНИЙ БАЙТ А0-А7
627	024F	C2 58 02		JNZ	PROG1	
628	0252	7A		MOV	A,D	
629	0253	FE 03		CPI	03	ПОСЛЕДНИЙ БАЙТ А8-А9
630	0255	CA 5D 02		JZ	LORC	ПОСЛЕДНИЙ АДРЕС(ДА)
631	0258	23	PROG1	INX	H	
632	0259	13		INX	D	УВЕЛИЧИТЬ АДРЕСА ОЗУ И ПРОГ. ПЗУ
633	025A	C3 35 02		JMP	PROG2	ОБРАБОТАТЬ СЛЕДУЮЩИЙ АДРЕС
634	025D		*			
635	025D		*			ПОСЛЕДНИЙ РАЗ ОБРАБАТЫВАЛИСЬ ВСЕ 1024 АДРЕСА?
636	025D		*			
637	025D	3A 06 10	LORC	LDA	CNT1	ЗАГРУЗИТЬ СЧЕТЧИК
638	0260	3C		INR	A	
639	0261	32 06 10		STA	CNT1	
640	0264	FE C8		CPI	200	200 ЦИКЛОВ??
641	0266	C2 2F 02		JNZ	PROG3	НЕ 200 ЦИКЛОВ,ПОВТОРНЫЙ СТАРТ
642	0269		*			
643	0269		*			ПРОГРАММИРОВАНИЕ ОКОНЧЕНО,ТЕПЕРЬ ПРОВЕРИТЬ
644	0269		*			
645	0269	3E 01		MVI	A,01	ЗАПРЕТИТЬ ПЕРЕДАЧУ ДАННЫХ В ПР.ПЗУ
646	026B	D3 32		OUT	32H	
647	026D	3D		DCR	A	АККУМУЛЯТОР=0
648	026E	D3 35		OUT	35H	CS/WE = 0.0 V
649	0270		*			
650	0270		*			ИНИЦИАЛИЗИРОВАТЬ АДРЕС ПРОГ. ПЗУ И ОЗУ
651	0270		*			
652	0270	11 00 00		LXI	D,00	
653	0273	21 78 05		LXI	H,1400	АДРЕС ОЗУ
654	0276		*			
655	0276	7B	PROG4	MOV	A,E	А0-А7 В АККУМУЛЯТОР
656	0277	D3 33		OUT	33H	АДРЕС В ПРОГ. ПЗУ
657	0279	7A		MOV	A,D	А8-А9 В АККУМУЛЯТОР
658	027A	D3 34		OUT	34H	А8-А9 В ПРОГ. ПЗУ
659	027C	DB 31		IN	31H	ЧИТАТЬ ДАННЫЕ ИЗ ПРОГ.ПЗУ В АККУМ.

```

660 027E BE      CMP M      СРАВНИТЬ ДАННЫЕ В ОЗУ И ПРОГ.ПЗУ
661 027F C2 93 02 JNZ PROG5 НЕ РАВНЫ,ОШИБКА
662 0282      *
663 0282      *
664 0282      *
665 0282      * АДРЕС ПОСЛЕДНИЙ
666 0282      *
667 0282 7B      MOV A,E
668 0283 FE FF    CPI OFFH - ПОСЛЕДНИЙ АО-А7?
669 0285 C2 8E 02 JNZ PROG5 НЕ ПОСЛЕДНИЙ
670 0288 7A      MOV A,D      ПОСЛЕДНИЙ А8-А9
671 0289 FE 03    CPI 03
672 028B CA 03 00 JZ BEGIN   ПОСЛЕДНИЙ.НАЧАТЬ НОВУЮ КОМАНДУ
673 028E      *
674 028E      * УВЕЛИЧИТЬ АДРЕСА ОЗУ И ПРОГ.ПЗУ
675 028E      *
676 028E 23      PROG5 INX H
677 028F 13      INX D
678 0290 C3 76 02 JMP PROG4   СРАВНЕНИЕ ПО СЛЕДУЮЩИМ АДРЕСАМ
679 0293      *
680 0293      * БЛОК ОБРАБОТКИ ОШИБОК
681 0293      *
682 0293 7B      PROG5 MOV A,E      АО-А7 В АККУМУЛЯТОР
683 0294 D3 F0    OUT OF0H
684 0296 7A      MOV A,D      А8-А9 В АККУМУЛЯТОР
685 0297 D3 F1    OUT OF1H    НА ДИСПЛЕЙ
686 0299 DB 31    IN 31H      ПРИНЯТЬ ДАННЫЕ ИЗ ПРОГ. ПЗУ
687 029B D3 F2    OUT OF2H    НА ДИСПЛЕЙ
688 029D      *
689 029D      * ЗАЩИТИТЬ ПРОГ.ПОКА НЕ БУДЕТ НАЖАТА КЛАВИША CF
690 029D      *
691 029D CD 64 00 PROG6 CALL KDET
692 02A0 3A 02 00 LDA KWGHT
693 02A3 FE 16    CPI 16H      НАЖАТА КЛАВИША CF??
694 02A5 C2 9D 02 JNZ PROG6
695 02AB C3 03 00 JMP BEGIN   ВЕРНУТЬСЯ И ОБРАБОТАТЬ ДРУГУЮ КОМАН.
696 02AB      *
697 02AB      * *****
698 02AB      *
699 02AB      * НА РИС.9.21 ПОКАЗАНА ПРОГРАММА VFY ,БЛОК-СХЕМА
700 02AB      * КОТОРОЙ ИЗОБРАЖЕНА НА РИС.9.20
701 02AB      *
702 02AB      * *****
703 02AB      *
704 02AB      *
705 02AB      * УСТАНОВИТЬ РЕЖИМ ЧТЕНИЯ ИЗ ПРОГ.ПЗУ
706 02AB      *
707 02AB AF      VFY XRA A
708 02AC 3C      INR A
709 02AD D3 32    OUT 32H      ОБЕСПЕЧИТЬ РЕЖИМ ЧТЕНИЯ
710 02AF AF      XRA A
711 02B0 D3 35    OUT 35H      CS=0.0 V,PGM=0.0V
712 02B2 D3 33    OUT 33H      АДРЕС АО-А7=0
713 02B4 D3 34    OUT 34H      АДРЕС А8-А9=0
714 02B6 11 00 00 LXI D,0000   УСТАНОВИТЬ АДРЕС ПРОГ.ПЗУ
715 02B9 21 00 14 LXI H,1400H   УСТАНОВИТЬ АДРЕС ОЗУ
716 02BC      *
717 02BC      * НАЧАТЬ ЦИКЛ ЧТЕНИЯ И ПРОВЕРКИ
718 02BC      *
719 02BC DB 31    VFY IN 31H      ЧИТАТЬ ДАННЫЕ ИЗ ПРОГ.ПЗУ

```

720 02BE BE	CMP	M	СРАВНИТЬ ДАННЫЕ ОЗУ И ПРОГ. ПЗУ
721 02BF C2 DC 02	JNZ	V FALL	ЕСЛИ НЕ РАВНЫ, ТО НА V FALL
722 02C2 3E 13	MVI	A, 13H	
723 02C4 BA	CMP	D	ПРОВЕРИТЬ СТАРШИЙ БАЙТ, ПОСЛЕД. АДРЕС
724 02C5 C2 D1 02	JNZ	V NEX	НЕ ПОСЛЕДНИЙ, ПРИНЯТЬ СЛЕДУЮЩИЙ АДРЕС
725 02C8 3E FF	MVI	A, OFFH	
726 02CA BB	CMP	E	ПРОВЕРИТЬ МЛАДШИЙ БАЙТ, ПОСЛЕД. АДРЕС
727 02CB C2 D1 02	JNZ	V NEX	НЕ ПОСЛЕДНИЙ, ПРИНЯТЬ СЛЕДУЮЩИЙ АДРЕС
728 02CE C3 03 00	JMP	BEGIN	НАЗАД К ОСНОВНОЙ ПРОГРАММЕ
729 02D1 13	INX	D	
730 02D2 7A	MOV	A, D	НОВЫЙ АДРЕС ПРОГ. ПЗУ
731 02D3 D3 34	OUT	34H	ВЫДАТЬ ДАННЫЕ
732 02D5 7B	MOV	A, E	
733 02D6 D3 33	OUT	33H	ВЫДАТЬ ДАННЫЕ
734 02D8 23	INX	H	НОВЫЙ АДРЕС ОЗУ
735 02D9 C3 BC 02	JMP	V F Y 1	СРАВНИТЬ СЛЕДУЮЩИЕ ДАННЫЕ
736 02DC D3 F0	V FALL	OUT OF ON	ВЫДАТЬ НА ДИСПЛЕЙ ---ХХ
737 02DE 7B	MOV	A, E	
738 02DF D3 F1	OUT	OF 1H	
739 02E1 7A	MOV	A, D	
740 02E2 D3 F2	OUT	OF 2H	ВЫВОД АДРЕСА НА ДИСПЛЕЙ
741 02E4 CD 64 00	V OUT	CALL KDET	ЖДАТЬ НАЖАТИЯ КЛАВИШИ
742 02E7 3A 02 10	LDA	KW GHT	
743 02EA FE 16	CPI	16H	НАЖАТА КЛАВИША СЕТ
744 02EC CA 03 0H	JZ	BEGIN	
745 02EF C2 E4 02	JNZ	V OUT	ИДТИ НА ОЖИДАНИЕ НАЖАТИЯ КЛАВИШИ
746 02F2	*		
747 02F2	*		
748 02F2	*****		
749 02F2	*		
750 02F2	*	НА РИС.9.23 ПОКАЗАНА ПРОГРАММА COPY, БЛОК-СХЕМА	
751 02F2	*	КОТОРОЙ ИЗОБРАЖЕНА НА РИС.9.22	
752 02F2	*		
753 02F2	*		
754 02F2	*****		
755 02F2	*		
756 02F2	*		
757 02F2	*	РЕЖИМ ЧТЕНИЯ ИНФОРМАЦИИ ИЗ ПРОГ. ПЗУ	
758 02F2	*		
759 02F2 AF	COPY	XRA A	
760 02F3 3C	INR	A	АККУМУЛЯТОР=1
761 01F4 D3 32	OUT	32H	
762 02F6 AF	XRA	A	
763 02F7 D3 35	OUT	35H	CS=0.0V, PGM=0.0V
764 02F9 11 00 00	LXI	D, 0000	АДРЕС ПРОГ. ПЗУ=0
765 02FC 21 00 14	LXI	H, 1400H	УСТАНОВИТЬ АДРЕС ОЗУ
766 02FF 7B	COPY 1	MOV A, E	
767 0300 D3 33	OUT	33H	АДРЕС A0-A7
768 0302 7A	MOV	A, D	
769 0303 D3 34	OUT	34H	АДРЕС A8-A9
770 0305 DB 31	IN	31H	ЧИТАТЬ ИНФОРМАЦИЮ ИЗ ПРОГ. ПЗУ
771 0307 77	MOV	M, A	ВЫВЕСТИ ИНФОРМАЦИЮ В ПЗУ
772 0308 3E 13	MVI	A, 13H	
773 030A BA	CMP	D	ПРОВЕРКА НА ПОСЛЕДНИЙ АДРЕС
774 030B C2 14 03	JNZ	C NEX	ЕСЛИ АДРЕС НЕ ПОСЛЕДНИЙ, ТО НА C NEX
775 030E 3E FF	MVI	A, OFFH	
776 0310 BB	CMP	E	МЛАДШИЙ БАЙТ ПОСЛЕДНЕГО АДРЕСА
777 0311 CA 03 00	JZ	BEGIN	ЕСЛИ АДРЕС ПОСЛЕДНИЙ, ТО КОНЕЦ
778 0314 23	C NEX	INX H	СЛЕДУЮЩИЙ АДРЕС ОЗУ
779 0315 13	INX	D	СЛЕДУЮЩИЙ АДРЕС ПРОГ. ПЗУ
780 0316 C3 FF 02	JMP	COPY 1	ПРИНЯТЬ СЛЕДУЮЩИЕ ДАННЫЕ
781 0319	*		
782 0319	END		ПРЕДЛОЖЕНИЕ END АССЕМБЛЕРА

0 ERRORS 60 SYMBOLS

совпадают для всех адресов, производится возврат в основную программу (шаг 7). На рис. 9.21 приведена реализация функции верификации средствами символического языка микропроцессора 8085.

9.9. Программные средства реализации функции копирования

Последней функцией, которую мы рассмотрим, является функция копирования, осуществляющая электрическое копирование данных из ППЗУ в ячейки ОЗУ₂. Эта функция используется для размножения уже существующего ППЗУ. Блок-схема реализации функции приведена на рис. 9.22. Из этой схемы видно, что на шаге 2 производится чтение данных из ППЗУ, а на шаге 3 — их запись в ОЗУ. Подобные действия повторяются для всего диапазона адресов 0000—03FF₁₆. На рис. 9.23 приведена реализация функции копирования средствами символического языка микропроцессора 8085.

9.10. Выводы по программным средствам

В настоящей главе были рассмотрены все основные аспекты проектирования и написания программ для системы, управляемой с помощью микропроцессора. С целью более подробного изложения частных вопросов был рассмотрен конкретный пример. При этом были раскрыты общие вопросы, характерные для различных задач. Для большей наглядности использовался аппарат блок-схем, а необходимая полнота изложения достигалась с помощью примеров программной реализации приведенных блок-схем.

В главе был последовательно изложен процесс создания программного обеспечения системы. При этом правильный выбор отправной точки гарантирует успех. В главе заострялось внимание на тех важных вопросах, решение которых необходимо осуществить до начала разработки программного обеспечения. Правильное понимание рассмотренной задачи будет способствовать правильному использованию других микропроцессоров для решения конкретных задач.

На рис. 9.24 приведена полная программа управления системой для микропроцессора 8085.

Глава 10

ОТЛАДКА ТЕХНИЧЕСКИХ СРЕДСТВ СИСТЕМЫ

В настоящей главе будут рассмотрены различные концепции поиска неисправностей и отладки системы, управляемой микропроцессором на примере устройства программирования ППЗУ, описанного в гл. 8 и 9. При этом используется техника поиска неисправностей, аналогичная той, которая обсуждалась в гл. 5. Будет подробно показано, каким образом можно проверить функционирование технических средств конкретной системы.

Еще раз будут рассмотрены вопросы отладки технических средств системы с использованием метода, известного под названием тестирования посредством статических сигналов. Этот эффективный и сравнительно экономичный метод предполагает единый прямой подход к проблеме поиска неисправностей в аппаратуре. Вследствие простоты и некоторых других преимуществ метода по сравнению с динамическим тестированием при решении большинства задач поиска неисправностей рассмотрим еще раз общие принципы и применение тестирования посредством статических сигналов.

10.1. Тестирование посредством статических сигналов

В гл. 5 были освещены общие принципы тестирования посредством статических сигналов, а собственно метод использовался при изложении вопросов отладки технических средств исходной системы. В настоящей главе этот метод применяется при отладке аппаратуры устройства программирования ППЗУ. Еще до установки микропроцессора в системе использование данного метода позволяет убедиться, что память и все схемы ввода-вывода функционируют правильно.

На рис. 10.1 приведена схема используемого устройства тестирования посредством статических сигналов. Прежде чем перейти к подробному изложению вопросов отладки системы 8085 с помощью этого устройства, кратко осветим некоторые важные моменты, относящиеся собственно к используемому методу. В первую очередь, отметим простоту технических средств устройства тестирования. Такое устройство можно построить с помощью стан-

дартных логических схем. Как видно из рис. 10.1, для этого достаточно использовать лишь пять различных ИС.

Во-вторых, тестирование посредством статических сигналов чрезвычайно удобно в применении. Отметим, что устройство тестирования осуществляет управление системой точно так же, как и микропроцессор. Поэтому для использования устройства тестирования посредством статических сигналов (УТСС) необходимо лишь понять особенности управления техническими средствами системы со стороны микропроцессора.

В-третьих, УТСС можно приспособить к особенностям конкретной системы. Простота технических средств УТСС позволяет модифицировать это устройство с целью реализации специальной функции, присущей лишь конкретной системе. Пользователь может по желанию вносить изменения в УТСС. Об этом свидетельствует тот факт, что фирма CMS поставляет версии УТСС для любого из микропроцессоров, рассмотренных в книге.

10.2. Устройство тестирования посредством статических сигналов для микропроцессора 8085

Рассмотрим функционирование УТСС, показанного на рис. 10.1, детально проанализировав назначение каждого компонента схемы. На рис. 10.1 можно заметить переключатели SA_{15} — SA_8 . Каждый из них одним концом соединяется с землей, а другим — с согласующим резистором на 4,7 Ом, как это показано на рис. 10.1 лишь для переключателя SA_8 .

Переключатели расположены на входах интегральных схем IC_1 и IC_2 (рис. 10.1), представляющих собой устройства 7404. Выходами IC_1 и IC_2 являются адресные выходы A_{15} — A_8 для микропроцессора 8085. При разомкнутых переключателях входы IC_1 и IC_2 — в состоянии логической 1, которое является результатом воздействия источника питания в 5 В через согласующий резистор 4,7 кОм. При замыкании переключателей входы IC_1 и IC_2 заземляются, что соответствует сигналу логического 0.

Функционирование переключателей SA_{15} — SA_8 может быть в общем виде представлено следующим образом. В одном положении переключателей на соответствующие выходы IC_1 или IC_2 поступает сигнал уровня логической 1, в другом положении — сигнал уровня логического 0. Таким образом выходные линии A_{15} — A_8 можно по желанию устанавливать в состояние 0 или 1. При этом сигнал в систему поступает точно так же, как сигнал, выдаваемый микропроцессором 8085 на адресные линии, за исключением того, что соответствующий уровень логического сигнала для обеспечения отладки системы может подерживаться произвольно долго.

Рассмотрим теперь блок вывода данных УТСС. Этот блок — фрагмент рис. 10.1 и отдельно представлен на рис. 10.2. Отме-

тим, что переключатели S_0 — S_7 подсоединены точно так же, как ранее рассмотренные переключатели SA_{15} — SA_8 , т. е. размещены на входе устройств 74LS367 IC_3 и IC_4 . Выбор этих интегральных схем обусловлен использованием некоторых устройств с тремя состояниями для вывода данных. Необходимость по-

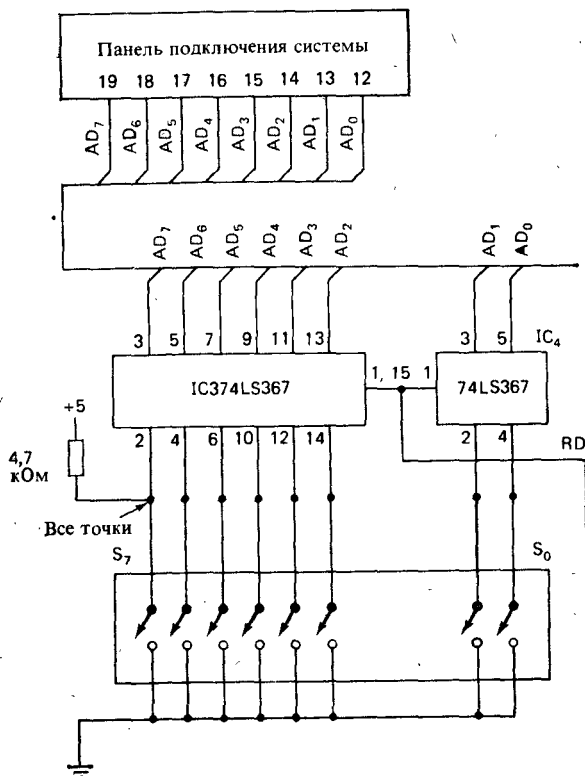


Рис. 10.2. Часть схемы рис. 10.1, отображающая блок вывода данных УТСС 8085.

следних связана с обеспечением возможности ввода данных в УТСС, что аналогично вводу в микропроцессор 8085, когда линии данных выступают в качестве входных. Поэтому следует заблокировать выдачу данных УТСС формирователями IC_3 и IC_4 на время операции чтения, выполняемой УТСС, чтобы не создавать помех при вводе.

Блокировка выдачи данных формирователями осуществляется посредством управляющего сигнала RD. Линия подачи этого сигнала показана справа на рис. 10.2. Ниже будет показано, как этот сигнал вырабатывается. Теперь же важно понять лишь то, что выдача данных формирователями на время выполнения УТСС операции чтения блокируется.

Во время выполнения УТСС операции чтения данных вводятся в УТСС из тестируемой системы. Для проверки правильности вводимых данных используются индикаторы на светоизлучающих диодах. Схема включения этих индикаторов приведена на рис. 10.3. Отметим, что все входы устройств 74L04 IC₈ и IC₉ подсоединены к отдельным линиям шины данных. Логи-

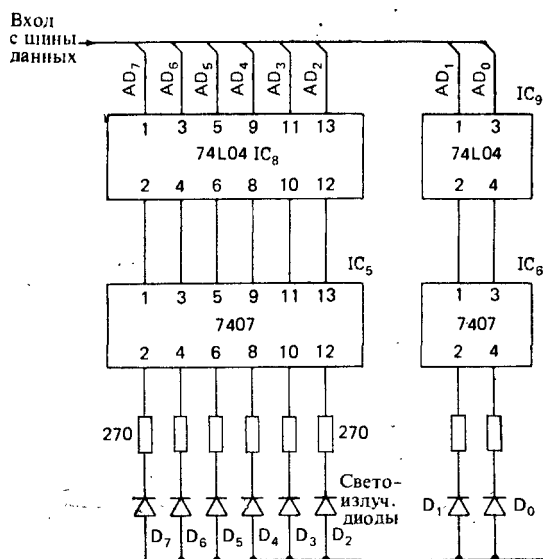


Рис. 10.3. Часть схемы рис. 10.1, отображающая блок вывода сигналов на светоизлучающие диоды УТСС 8085.

ческие сигналы линий данных отображаются посредством показанных на рис. 10.3 светоизлучающих диодов.

Если на вход IC₈ и IC₉ подается сигнал 1, то на соответствующем выходе устройства 74L04 формируется сигнал 0. Выход 74L04 соединен со входом 7407 (IC₅ и IC₆), являющимся буфером со свободным коллектором. В случае сигнала 0 на входе 7407 на выходе также присутствует сигнал 0. Из теории устройств со свободным коллектором известно, что выход устройства 7407 может быть представлен в виде, изображенном на рис. 10.4.

Ясно, что если на вход 7407 поступает сигнал 0, то транзистор открывается, в результате от коллектора к эмиттеру на землю протекает ток. Обращаясь снова к рис. 10.3, видим, что светоизлучающий диод анодом подключен к источнику +5 В, а катодом — к выходу устройства 7407 через резистор 270 Ом. Если на выходе устройства 7407 имеется сигнал 0, резистор соединяется с землей, в результате через светоизлучающий диод

от анода к катоду протекает ток от источника $+5$ В. При этих условиях последний имеет прямое смещение и излучает свет. Если же на выходах устройств 7407, IC_5 и IC_6 , имеется сигнал 1, транзистор на рис. 10.4 закрывается, в результате происходит разьединение с землей резистора, подсоединенного к коллектору. При этих условиях ток через светоизлучающий диод не протекает и последний выключается.

Из проведенного обсуждения видно, что светоизлучающий диод включается при сигнале уровня логической 1 на соответ-

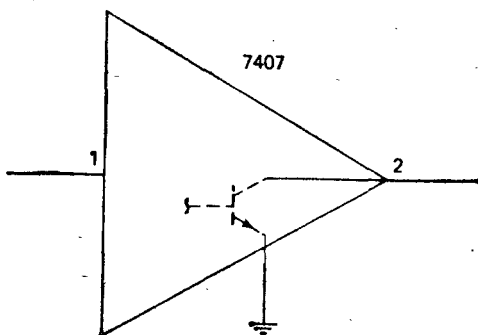


Рис. 10.4. Схема устройства 7407, показывающая выходной транзистор, используемый для отвода тока при включении светоизлучающих диодов.

ствующей линии шины данных и выключается при сигнале уровня логического 0. Это позволяет визуально наблюдать логическое состояние шины данных в любой момент времени.

Отметим, что в УТСС на входе с шины данных имеется устройство 74L04, что видно из рис. 10.3. Может возникнуть вопрос, почему на входе включается инвертор и далее устройство 7407 и нельзя ли применить устройство 7406 для реализации функций устройств 74L04 и 7407? Принятое решение мотивируется тем, что устройство 74L04 потребляет на входе ток $0,1$ мА в состоянии логического 0 и 4 мкА — в состоянии логической 1. Именно такой ток потребляется при работе с шиной данных.

Любое устройство, используемое в УТСС, является дополнительной нагрузкой для шины данных. Поэтому дополнительная нагрузка, создаваемая монитором, должна быть по возможности минимальной. Устройство 7406, используемое в качестве входного монитора, создает для шины данных большую нагрузку, чем устройство 74L04, поскольку потребляет на входе ток $1,6$ мА в состоянии логического 0 и 40 мА в состоянии логической 1. Поэтому применение устройств 74L04 и 7407 вместо 7406 позволяет снизить нагрузку на шину со стороны УТСС на порядок, что вполне оправдывает схемную избыточность.

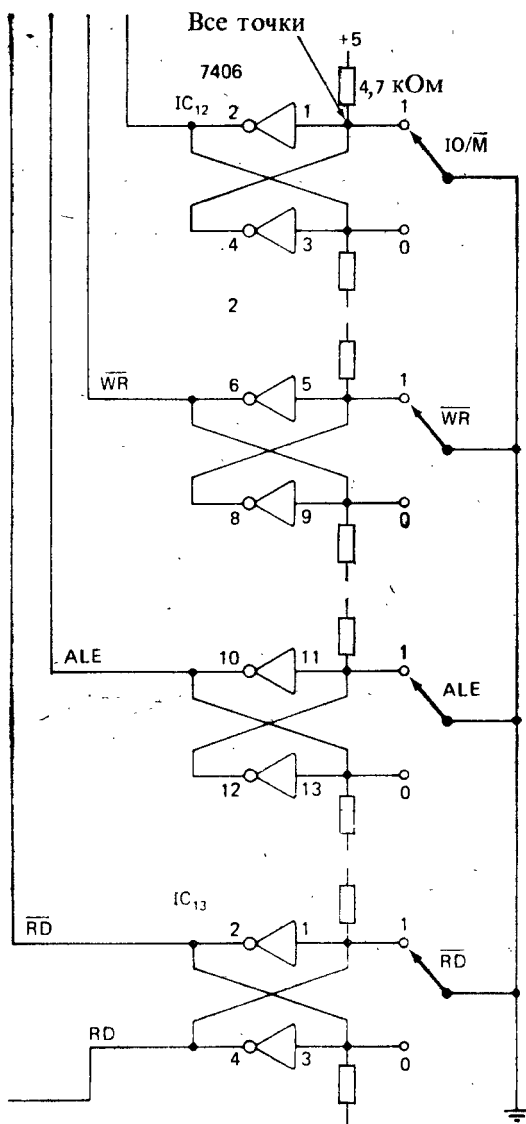


Рис. 10.5. Часть схемы рис. 10.1, отображающая блок УТСС, предназначенный для регулирования логического уровня управляющих разрядов микропроцессора 8085.

Из рис. 10.3 видно, что светоизлучающие диоды постоянно контролируют и отображают логическое состояние линий шины данных, как во время выполнения операции ЧТЕНИЕ, так и при операции ЗАПИСЬ. При операции ЧТЕНИЕ отображаются

«системные» данные, при операции ЗАПИСЬ — данные, выводимые УТСС. Это возможно, поскольку выходные формирователи IC_3 и IC_4 блокируются на время операции ЧТЕНИЕ.

Рассмотрим, каким образом осуществляется блокировка этих формирователей при выполнении операции ЧТЕНИЕ. На рис. 10.5 показана секция устройства УТСС, используемая для регулирования логических уровней сигналов некоторых управляющих выводов микропроцессора 8085. Эти выводы обозначены как $\overline{RD}$, $\overline{WR}$, $IO/\overline{M}$, ALE . Существуют другие управляющие

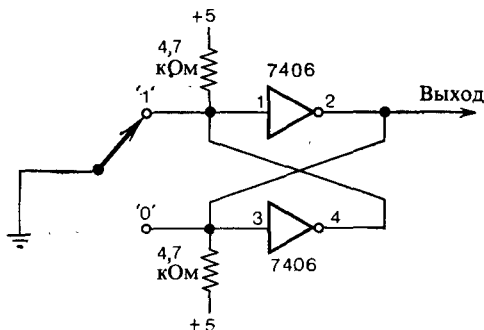


Рис. 10.6. Построение схемы гашения «выброса» с помощью двух инверторов со свободным коллектором типа 7406.

выводы микропроцессора 8085, такие, как S_0 и S_1 ; однако они не используются в нашей системе. Если же в проектируемой системе используются другие управляющие выводы, то для регулирования логических уровней состояний УТСС следует применить подобные же средства.

Рассмотрение событий, происходящих в течение выполнения операции ЧТЕНИЕ микропроцессора 8085, показывает, что одним из основных событий является выдача сигнала $\overline{RD}$. При переходе этого сигнала в состояние, соответствующее логическому 0, микропроцессор выполняет операцию чтения либо из памяти, либо с устройства ввода, что идентифицируется логическим состоянием сигнала $IO/\overline{M}$. По этой информации УТСС определяет необходимость блокировки выходных формирователей IC_3 и IC_4 .

Устройство 74LS367 блокируется, если на выводы 1 и 15 поступает сигнал уровня логической 1. Следовательно, в случае когда сигнал соответствует логическому 0, на эти выводы устройства 74LS367 необходимо подать сигнал 1. Это достигается путем подачи на блокирующие выводы 1 и 15 схем IC_3 и IC_4 , приведенных на рис. 10.2, сигнала $\overline{RD}$, как это показано на рис. 10.5. Соответствующее соединение осуществляется всякий раз, когда сигнал $\overline{RD}$ есть 0, что определяет операцию ЧТЕНИЕ

для УТСС микропроцессора 8085. При этом светонизлучающие диоды отображают данные, вводимые в микропроцессор.

Управляющие сигналы УТСС поступают на фиксатор, как это показано на рис. 10.6, используемый с целью предотвращения «выброса» сигнала, обусловленного действием механического переключателя. Может возникнуть вопрос, почему бы не поставить подобные схемы для каждого переключателя? Это не делается по той причине, что адресные линии и линии данных микропроцессора не оказывают влияния на коммуникационные взаимодействия в системе до тех пор, пока эти взаимодействия не распознаны посредством управляющих сигналов. Именно с помощью управляющих сигналов и осуществляется стробирование данных, передаваемых во внешние схемы. Для каждого положения переключателя требуется один стробирующий сигнал. Схемы, предотвращающие выброс, можно, по желанию, установить для каждого переключателя, однако это не является необходимым.

10.3. Замечания по отладке технических средств системы

Рассмотрев особенности схемной реализации УТСС, изложим в общих чертах один подход, используемый для эффективной отладки технических средств системы. Существо подхода заключается в определении правильности функционирования всех коммуникационных каналов в системе. При этом не принимаются во внимание вопросы синхронизации сигналов, а определяется возможность электрического взаимодействия микропроцессора со всеми логическими устройствами в системе. Подобная проверка имеет большое значение для отладки технических средств системы.

Тестирование посредством статических сигналов предоставляет средства для осуществления проверки всех коммуникационных каналов системы. Это является большим достоинством данного метода, поскольку отпадает необходимость в проектировании специальных схем тестирования в рамках специфических тестирующих устройств микропроцессорной системы. Кроме того, основные средства устройства тестирования могут быть реализованы технически при минимальных затратах. Это дает возможность каждому пользователю, занимающемуся проектированием системы на базе микропроцессора либо поиском в ней неисправностей, иметь мощные средства отладки системы.

При использовании УТСС последнее подсоединяется к системе посредством кабеля, как показано в гл. 5. Последовательно рассмотрим проверку всех технических средств системы. При этом УТСС используем как средство отладки системы на

базе микропроцессора 8085. Однако приводимого материала достаточно для построения УТСС для проверки системы на базе любого другого микропроцессора.

10.4. Отладка адресной шины системы

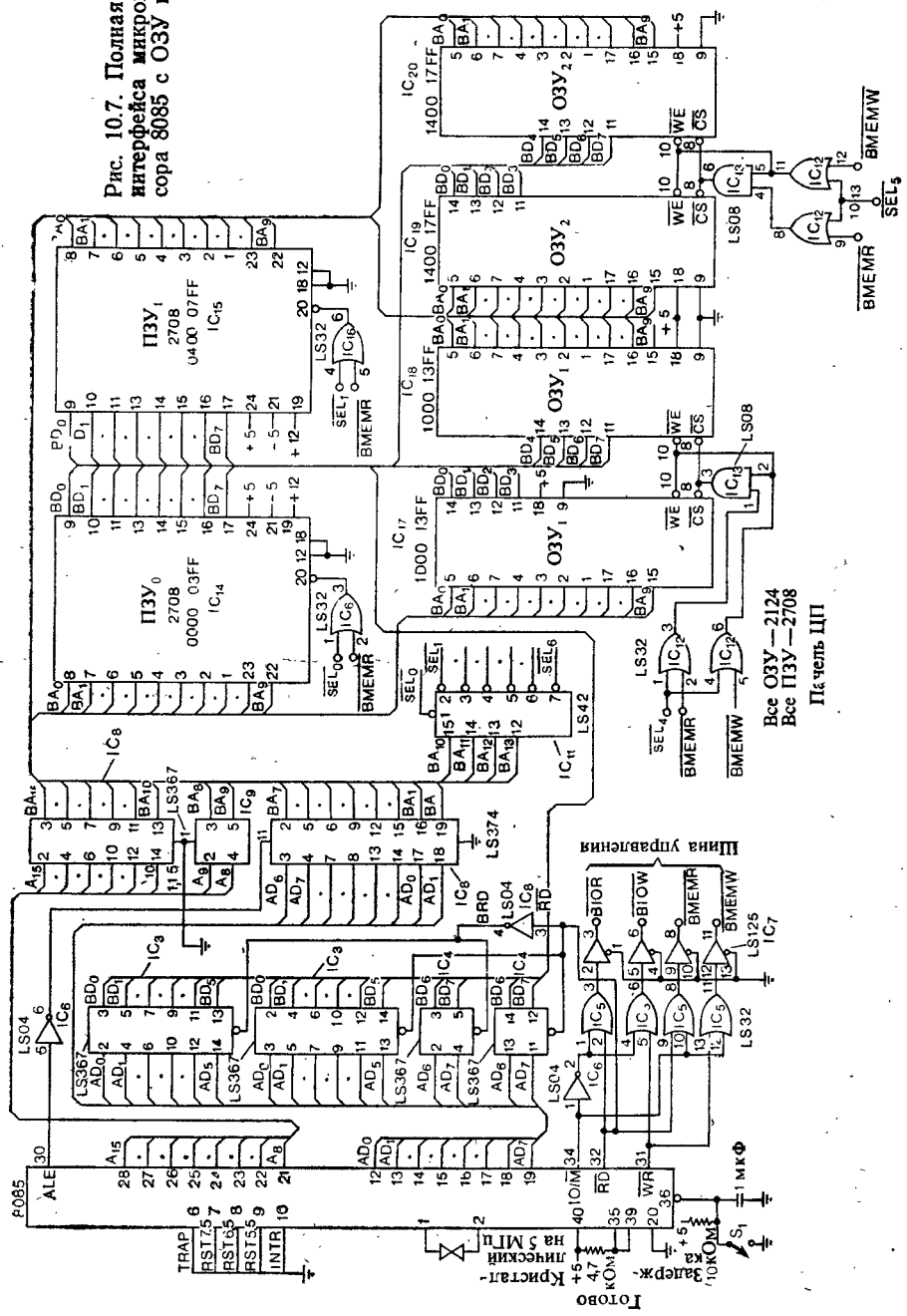
Полная схема интерфейса микропроцессора 8085 с ПЗУ и ОЗУ приведена на рис. 10.7. В последующем мы будем достаточно часто обращаться к рис. 10.7. Поэтому желательно каким-либо образом пометить этот рисунок, например с помощью бумажной полоски. Это позволит быстро его отыскивать каждый раз, когда в этом будет возникать необходимость.

Адресная шина системы — это первый компонент системы программирования ППЗУ, для которой мы будем осуществлять проверку прохождения электрических сигналов. Определим, возможна ли передача сигналов логических 0 и 1 по адресным линиям BA_0 — BA_{15} . С этой целью введем представленный на рис. 10.1 УТСС в проверяемую систему вместо микропроцессора путем установки вилки УТСС в гнездо подключения микропроцессора. Проще всего проверить адресные линии BA_8 — BA_{15} , поскольку последние в отличие от адресных линий BA_0 — BA_7 не имеют «временного мультиплексирования».

Процедура проверки линий BA_8 — BA_{15} посредством УТСС

1. Установить переключатели УТСС, помеченные как SA_8 — SA_{15} , в положение, соответствующее логическому 0 на выходе.
2. Соединить с землей один конец пробника логического уровня DVM (или VOM).
3. Соединить другой конец пробника напряжения с выводом 5 IC_9 системы (рис. 10.7).
4. Напряжение должно соответствовать логическому уровню 0.
5. Переключить SA_8 УТСС в положение, соответствующее логической 1.
6. Напряжение на выводе 5 IC_9 (рис. 10.7) должно соответствовать уровню логической 1.
7. Теперь обеспечено переключение адресной линии BA_8 с уровня логического 0 на уровень логической 1.
8. Повторить шаги 3—7 для каждой адресной линии BA_9 — BA_{15} , проверяя наличие требуемых уровней напряжения на соответствующих выходах устройства.
9. В случае отсутствия переключения сигнала на адресной линии статически исследуется причина неполадки.
10. Теперь проверено, что для всех адресных линий BA_8 — BA_{15} производится переключение сигнала с уровня 0 на уро-

Рис. 10.7. Полная схема интерфейса микропроцессора 8085 с ОЗУ и ПЗУ.



Всё ОЗУ — 2124
Всё ПЗУ — 2708
Память ЦП

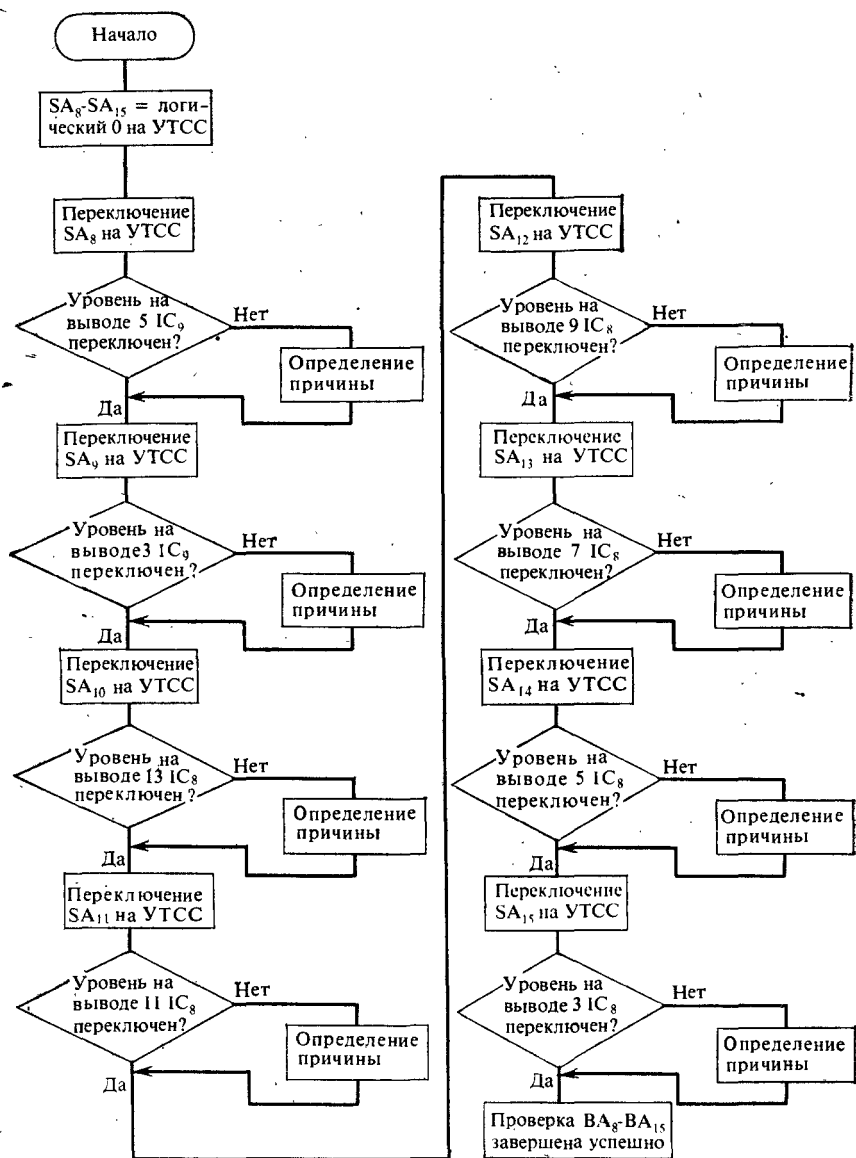


Рис. 10.8. Блок-схема алгоритма проверки выходов BA_8 — BA_{15} адресной шины микропроцессора 8085.

вень 1 под управлением микропроцессора; либо обнаружено, что для некоторых линий такое переключение отсутствует, что требует дальнейшего исследования (с этой целью осуществляется проверка каждой цепи схемы путем прослеживания уровня напряжения в выбранных точках на протяжении всего информационного канала).

На рис. 10.8 приведена блок-схема последовательности действий при проверке адресных линий BA_8 — BA_{15} . Вспомним, что мы не определяли правильность подсоединения линий BA_8 — BA_{15} в системе. Этот вопрос будет затронут в данной главе позднее.

Определим теперь возможность переключения линий BA_0 — BA_7 с уровня логической 1 на уровень логического 0. Соответствующая процедура несколько сложнее процедуры для адресных линий BA_8 — BA_{15} .

Процедура для проверки адресных линий BA_0 — BA_7

1. Установить переключатели, помеченные S_0 — S_7 , в положение, соответствующее логическому 0.

2. Установить управляющие переключатели УТСС следующим образом:

а) $\overline{WR}$ = логическая 1 (нерабочее состояние);

б) $\overline{RD}$ = логическая 1 (нерабочее состояние);

в) $IO/\overline{M}$ = логическая 1 (безразлично);

г) ALE = логический 0 (нерабочее состояние).

3. При сигнале $\overline{RD}$, соответствующем логическому 0, светозлучающие диоды будут (должны) отображать информацию, установленную с помощью переключателей S_0 — S_7 .

4. Установить $\overline{RD}$ в положение, соответствующее логической 1. При этом светозлучающий диод D_0 должен начать светиться.

5. Перебросить управляющий переключатель ALE из состояния 0 в состояние 1.

6. Установить управляющий переключатель ALE обратно в состояние 0. Последовательность шагов 5 и 6 приводит к фиксации адресных разрядов.

7. Измерить напряжение на выводе 10 IC. Это напряжение должно соответствовать значению уровня логической 1.

8. Установить S_0 в положение, соответствующее логической 1.

9. Повторить шаги 4—8 для каждой адресной линии BA_0 — BA_7 . Если на шаге 7 выходной сигнал не соответствует введенным данным, повторить процедуру с шага 3 и измерить напряжения на каждом шаге с целью проверки логической корректности сигналов.

Нашей целью является реализация общей функции «фикса-

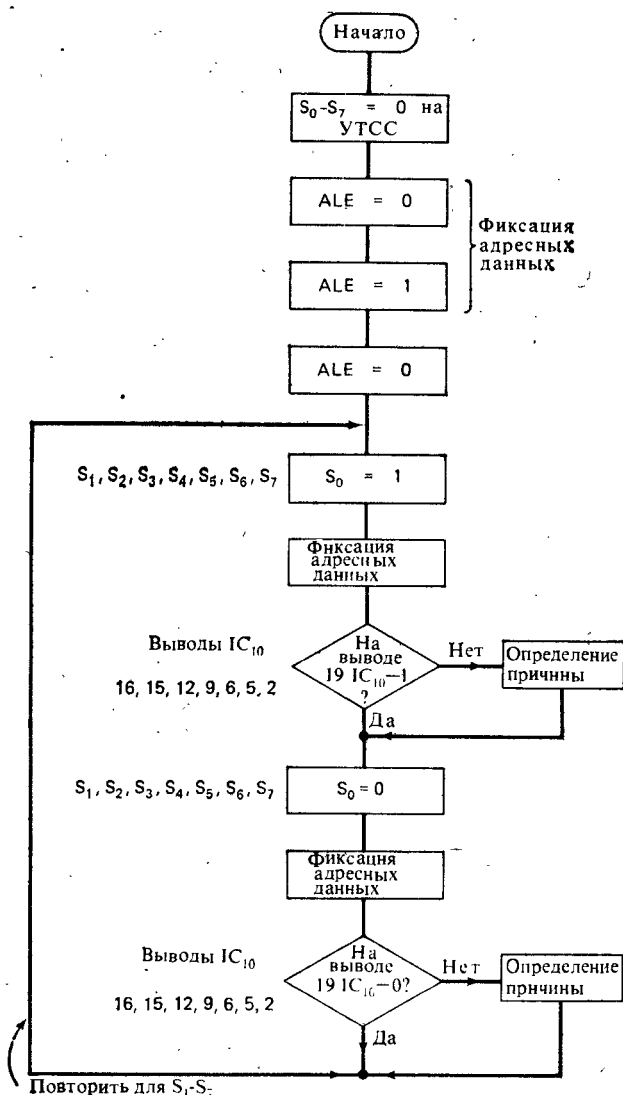


Рис. 10.9. Блок-схема алгоритма проверки выходов BA₀—BA₇ адресной шины.

ции» адресных данных. В случае неверной реализации функции мы можем разложить эту функцию на «микро-шаги» с целью локализации неисправности. На рис. 10.9 приведена блок-схема, отражающая основные этапы проверки адресных линий BA₀—BA₇.

10.5. Отладка шины управления системы

Обратимся теперь к проверке с помощью УТСС шины управления системы. Шина управления представлена на схеме рис. 10.10, которая является частью общей схемы рис. 10.7, что сделано с целью облегчить последующее обсуждение. В дальнейшем следует еще раз изучить схему рис. 10.7, чтобы убе-

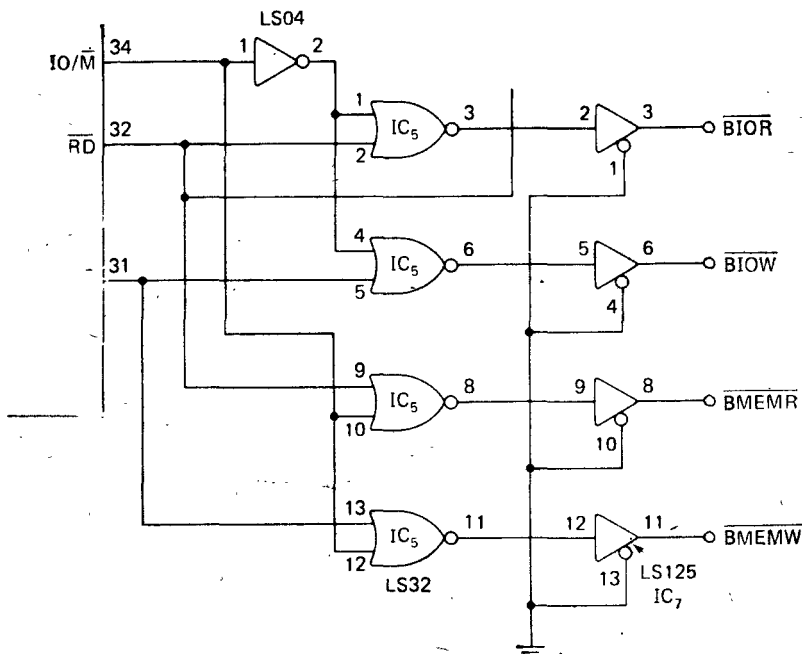


Рис. 10.10. Часть схемы рис. 10.7, определяющая шину управления системы.

даться в правильном понимании роли схемы шины управления, приведенной на рис. 10.10, в системе.

При настройке шины управления системы будем использовать управляющие переключатели УТСС, помеченные как $\overline{RD}$, $\overline{WR}$, $\overline{IO/M}$. Из рис. 10.10 видно, что эти три сигнала являются входами для комбинационной логической схемы, включающей шину управления системы. Управляющие сигналы вырабатываются вентилем ИЛИ схемы IC₅ и инвертором схемы IC₆. Перед вводом в систему эти сигналы буферизируются устройством 74LS125 IC₇. При настройке шины управления системы может быть использована следующая процедура.

Процедура настройки шины управления системы

1. Установить управляющие переключатели, помеченные как $\overline{RD}$, $\overline{WR}$, $IO/\overline{M}$, в следующие положения:

- а) $\overline{RD}$ = логическая 1 (нерабочее состояние);
- б) $\overline{WR}$ = логический 0 (нерабочее состояние);
- в) $IO/\overline{M}$ = логическая 1 (режим ввода-вывода).

2. При указанном положении переключателей на всех выходах шины управления системы должен быть сигнал уровня логической 1. Проверить наличие сигнала уровня логической 1 на выводах 3, 6, 8 и 11 устройства 74LS125 IC₇.

3. Установить в состояние 0 переключатель, помеченный как $\overline{WR}$, что соответствует подаче управляющего сигнала $\overline{BIOW}$. Проверить наличие напряжения логического уровня 0 на выводе 6 IC₇ ($\overline{BIOW}$).

4. Проверить наличие напряжения логического уровня 1 на выводах 3, 8 и 11 IC₇ (нерабочее состояние).

5. При проверке функционирования каждой линии шины необходимо помнить, что на основании результатов предыдущего обсуждения сигналы этих линий являются взаимоисключающими. Это означает, что одновременно сигнал не может быть подан ни на какие две линии.

6. Установить переключатель $\overline{WR}$ в положение, соответствующее логической 1.

7. Установить переключатель $\overline{RD}$ в положение, соответствующее логическому 0, что соответствует подаче управляющего сигнала $\overline{BIOR}$.

8. Проверить, что логический уровень напряжения 0 имеет единственный управляющий сигнал $\overline{BIOR}$ на выводе 3 IC₇.

9. Установить переключатель $\overline{RD}$ в положение, соответствующее логической 1.

10. Теперь проверена правильность прохождения сигналов $\overline{BIOW}$ и $\overline{BIOR}$.

11. Установить управляющий переключатель $IO/\overline{M}$ в положение, соответствующее логическому 0, что переводит шину управления в режим работы с памятью.

12. Установить переключатель $\overline{WR}$ в положение, соответствующее логическому 0, что соответствует подаче сигнала $\overline{BMEMW}$. Проверить наличие напряжения логического уровня 0 сигнала $\overline{BMEMW}$ на выводе 11 IC₇.

13. Проверить наличие напряжения логического уровня 1 сигналов $\overline{BIOR}$, $\overline{BIOW}$, $\overline{BMEMR}$.

14. Установить переключатель $\overline{WR}$ в положение, соответствующее логической 1.

15. Установить переключатель $\overline{RD}$ в положение, соответствующее логическому 0, что соответствует подаче сигнала $\overline{BMEMR}$. Проверить наличие напряжения логического уровня 0 сигнала $\overline{BMEMR}$ на выводе 8 IC₇.

16. Проверить наличие напряжения логического уровня 1 управляющих сигналов $\overline{BMEMW}$, $\overline{BIOW}$, $\overline{BIOR}$.

17. Установить переключатель $\overline{RD}$ в положение, соответствующее логической 1.

Теперь установлено, что выходные управляющие сигналы системы правильно отражают входные воздействия микропроцессора 8085. Если на каком-то шаге процедуры будет обнаружено нарушение этого соответствия, можно легко осуществить статическую проверку логических сигналов и локализовать неисправность.

10.6. Отладка шины данных системы

Рассмотрим теперь отладку шины данных системы с помощью УТСС. Приведем процедуру определения возможности двунаправленной передачи данных по шине данных, проверив буферирование шины данных и внешние устройства.

Процедура проверки канала выдачи данных

1. Установить управляющие переключатели $\overline{RD}$ и $\overline{WR}$ УТСС в положения:

а) $\overline{RD}$ = логическая 1;

б) $\overline{WR}$ = логическая 1.

При этих условиях существует канал из УТСС микропроцессора 8085 к тестируемой системе. УТСС функционирует в режиме вывода данных, аналогичном соответствующему режиму микропроцессора 8085.

2. Установить переключатели УТСС, помеченные как S_0 — S_7 , в положение, соответствующее логическому 0.

3. Установить переключатель S_0 в положение, соответствующее логической 1.

4. Проверить наличие напряжения логического уровня 1 на выходе BD_0 (вывод 2 IC₃).

5. Установить переключатель S_0 в положение, соответствующее логическому 0. Проверить наличие напряжения логического уровня 0 на выходе BD_0 (вывод 2 IC₃).

6. Повторить шаги 3—5 для всех линий шины данных DB_1 — DB_7 .

7. На каждом шаге проверить правильность функционирования линий шины данных при выводе информации путем измерения напряжения логических уровней.

Если выходные сигналы на шине данных на некотором шаге процедуры не являются корректными, необходимо статически проверить уровни напряжения постоянного тока с целью определить причины неисправности. Это достигается путем проверки статических электрических сигналов на протяжении информационного канала.

Успешное завершение процедуры означает, что шина данных системы при выводе данных функционирует правильно, т. е. шина данных должна выводить данные, соответствующие правильным сигналам от микропроцессора 8085. Теперь проверим правильность функционирования шины данных в режиме ввода и покажем, каким образом можно проверить информационный канал от тестируемой системы к УТСС, чтобы убедиться в правильности его работы.

Процедура проверки канала ввода шины данных

1. Установить переключатели УТСС, помеченные как S_0 — S_7 , в положение, соответствующее логическому 0.
2. Установить адресные линии системы BA_0 — BA_{15} в состояние 0.

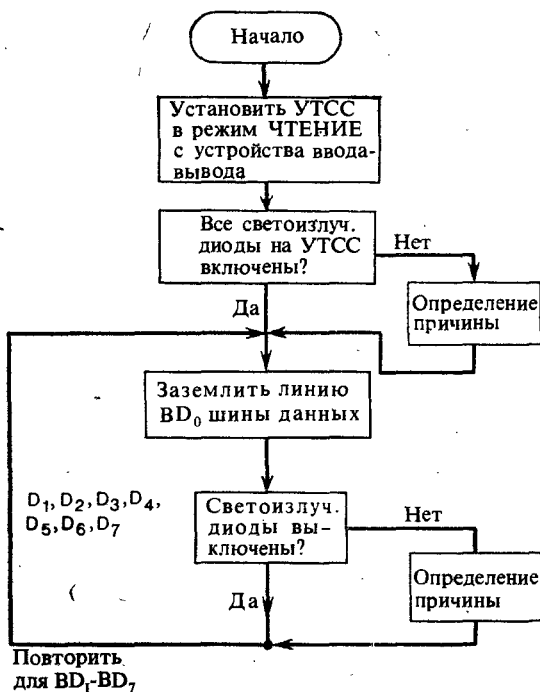


Рис. 10.11. Укрупненная блок-схема алгоритма проверки сигналов BD_1 — BD_7 шины данных системы.

3. Установить управляющие переключатели в положения:
 - а) $\overline{RD}$ = логическая 1;
 - б) $\overline{WR}$ = логическая 1;
 - в) $IO/\overline{M}$ = логическая 1 (режим ввода-вывода).
4. При указанном режиме управляющих переключателей светоизлучающие диоды выключены.
5. Установить переключатель $\overline{RD}$ в положение, соответствующее логическому 0. Теперь все светоизлучающие диоды должны зажечься, поскольку шина данных отключается. Тем самым никакое устройство вывода не управляет линиями шины данных. Входы устройств 74L04 IC₈ и IC₉ УТСС отключены, а отключение входа TTL-схемы соответствует логическому уровню 1.
6. Взять проволочную перемычку и заземлить один ее конец.
7. Соединить другой конец перемычки с линией BD_0 шины данных, что соответствует выводу 2 IC₃. Это позволяет подать на шину данных логический сигнал уровня 0 (заземление).
8. Проверить, что светоизлучающий диод УТСС, помеченный как D_0 , выключен.
9. Заземлить поочередно линии шины данных BD_1 — BD_7 , проверяя выключение соответствующих светоизлучающих диодов. Блок-схема описанной процедуры представлена на рис. 10.11.

Теперь рассмотрены процедуры проверки шин адресной, данных и управления системы. Эти шины будут правильно функционировать при правильном поступлении сигналов от микропроцессора 8085. Однако мы не осуществили проверок правильности соединения шин системы с другими схемами системы. Такие проверки будут выполнены в процессе дальнейшего обсуждения.

10.7. Проверка схем дешифрирования выбора памяти

Перейдем к проверке технических средств, предназначенных для дешифрирования адреса памяти. Логические схемы дешифрирования сигнала выбора памяти для системы, приведенной на рис. 10.7, представлены блоком IC₁₁, являющимся устройством 74LS42. Принцип действия устройства 74LS42 заключается в подаче выходного сигнала 0 на вывод, соответствующий каждому входу. При этом выходы 1, 2, 3, 4, 5, 6, 7 соответствуют двоичным входам 0, 1, 2, 3, 4, 5, 6 соответственно. Допустим, что код двоичного 0011 подан на выходы устройства 74LS42 12, 13, 14 и 15 соответственно (это соответствует двоичному входу 3). Вывод 4 устройства 74LS42 теперь находится в состоянии логического 0.

Процедура проверки устройства 74LS42

1. Установить переключатели УТСС, помеченные как SA_{10} — SA_{13} , в положение, соответствующее логическому 0. Отметим, исходя из рис. 10.7, что адресные входы BA_{10} — BA_{13} являются входами устройства 74LS42.

2. Проверить наличие напряжения логического уровня 0 на выводе IC_{11} .

3. Установить переключатели SA_{10} — SA_{13} в положения:

- а) $SA_{10}=1$;
- б) $SA_{11}=0$;
- в) $SA_{12}=0$;
- г) $SA_{13}=0$.

Проверить наличие напряжения логического уровня 0 на выводе 2 IC_{11} .

4. Установить переключатели УТСС SA_{10} — SA_{13} в положение двоичных входов 2—6. Значение каждого двоичного входа порождает логический уровень 0 для соответствующего вывода устройства 74LS42 IC_{11} . Необходимо проверить наличие логического уровня 0 для следующих выводов устройства 74LS42.

Двоичный вход	Уровень 0 на выводе
0010	2
0011	3
0100	4
0101	5
0110	6

Если все выводы имеют логический уровень 0 для всех вышеописанных шагов, можно считать, что правильность работы логических схем дешифрирования сигнала выбора памяти проверена. Если же на некотором шаге процедуры проверки обнаружена ошибка, ее причина может быть установлена путем статической проверки системы. Отметим, что правильность соединения линий выбора памяти пока еще не проверена. Соответствующая проверка осуществляется ниже.

10.8. Проверка интерфейса ПЗУ системы

При проверке интерфейса ПЗУ системы необходимо быть уверенным в правильности:

- 1) адреса ПЗУ;
- 2) данных, выдаваемых из ПЗУ;
- 3) информации выбора блока ПЗУ;
- 4) подачи питания на ПЗУ.

Обсуждение процедуры проверки интерфейса ПЗУ разбивается на четыре раздела. В каждом из разделов раскрывается один из перечисленных выше пунктов.

Проверка правильности адреса ПЗУ

Правильность адреса ПЗУ обуславливается корректным функционированием адресных входов ПЗУ $BA_0—BA_9$. Для их проверки может быть использована следующая процедура:

1. Установить адресные переключатели $SA_8—SA_{15}$ в положение, соответствующее логическому 0.

2. Установить управляющие переключатели $\overline{RD}$, $\overline{WR}$, $IO/\overline{M}$ в положения:

а) $\overline{RD}$ = логическая 1;

б) $\overline{WR}$ = логическая 1;

в) $IO/\overline{M}$ = логический 0.

3. Установить переключатели $S_0—S_7$ в положение, соответствующее логическому 0.

4. Установить переключатель S_0 в положение, соответствующее логической 1.

5. Поставить переключатель ALE в положение логической 1, а затем в положение логического 0. В результате этих действий фиксируются разряды адреса $BA_0—BA_7$ в устройстве 74LS374 IC₁₀.

6. Проверить наличие логического уровня напряжения 1 на выводе 8 IC₁₄ и IC₁₅ (BA_0).

7. Повторить шаги 3—6 для адресных входов $BA_1—BA_7$. Проверить наличие необходимого сигнала на соответствующем выводе устройства 2708 при подаче на адресную шину сигнала логической 1.

8. Проверить адресные входы ПЗУ $BA_8—BA_9$. Установить переключатель, помеченный как SA_8 , в положение, соответствующее логической 1. Проверить наличие сигнала уровня логической 1 на выводе 22 IC₁₄, IC₁₅ (BA_8). Повторить этот шаг для адресного входа BA_9 .

Если на каком-то шаге описанной процедуры получим неверный адрес ПЗУ IC₁₄—IC₁₅, такую ситуацию можно зафиксировать и статически проверить функционирование системы.

Проверка данных, считываемых из ПЗУ

При проверке данных, считываемых из ПЗУ, будем придерживаться следующей последовательности действий. Вначале система переводится в режим чтения данных, однако выбор устройства ввода не осуществляется. Существенным при этом является сам факт перевода шины данных системы в режим ввода, что позволяет использовать светоизлучающие диоды УТСС для визуального наблюдения за состоянием шины данных. Далее поочередно заземлим выводы данных ПЗУ с помощью проводника. При заземлении вывода ПЗУ соответствующий светоизлучающий диод выключается.

Отметим возможность электрического заземления выводов ПЗУ, что объясняется наличием трехстабильного режима этих выводов, который устанавливается при переключении системы в режим ЧТЕНИЯ с устройства ввода посредством шины управления системы. При осуществлении описываемой проверки определяется правильность физического соединения шины данных с выводами ПЗУ IC₁₄, IC₁₅. Проверка осуществляется следующей процедурой:

1. Установить переключатель IO/ $\overline{M}$ УТСС в положение, соответствующее вводу-выводу (в результате система переводится в режим ввода-вывода).

2. Установить переключатель $\overline{WR}$ в положение, соответствующее логической 1.

3. Установить переключатель $\overline{RD}$ в положение, соответствующее логическому 0. В результате система переводится в режим ЧТЕНИЯ с устройства ввода.

4. Установить системный адрес BA₀—BA₁₅=0000. В этом случае не выбирается ни одно устройство ввода.

5. Все светоизлучающие диоды в данном состоянии включены.

6. Заземлить поочередно выводы ПЗУ₀ и ПЗУ₁ с помощью проводника. Каждый раз при заземлении очередного вывода ПЗУ₀ или ПЗУ₁ соответствующий светоизлучающий диод УТСС выключается. Например, если заземлен вывод 8 IC₁₄ или IC₁₅ ПЗУ₀ или ПЗУ₁, то выключится светоизлучающий диод D₀ УТСС. Так осуществляется визуальная проверка электрической работоспособности информационного канала от выводов ПЗУ системы к микропроцессору 8085.

Проверка информации выбора блока ПЗУ

Покажем теперь, как проводится проверка правильности работы схем выбора блока ПЗУ. Как известно, функция операции выбора блока заключается в поочередной выдаче на шину данных системы выходной информации одного блока ПЗУ. Вначале рассмотрим, каким образом проверить выбор блока IC₁₄ ПЗУ.

Сигнал на выводе 20 выбора блока ПЗУ IC₁₄ принимает уровень логического 0 при сигнале логического 0 на выводе 3 IC₁₆. На выводе 3 сигнал уровня логического 0 появляется при сигналах 0 одновременно на выводе 1 ($\overline{SEL_0}$) и выводе 2 ($\overline{BMEMR}$). Это возможно, поскольку сигнал $\overline{BMEMR}$ принимает значение 0 при чтении данных из памяти, а сигнал $\overline{SEL_0}$ принимает значение 0, если системный адрес принадлежит соответствующему блоку ПЗУ IC₁₄.

Чтобы убедиться, что выбор блока осуществляется правильно, необходимо перевести систему в режим чтения данных из ПЗУ.

Процедура чтения данных из ПЗУ

1. Установить системный адрес $BA_0—BA_{15}$ в 0000. Из распределения памяти системы мы знаем, что ПЗУ₀ соответствует адресное пространство 0000—03FF. В результате генерируется логический уровень 0 сигнала $\overline{SEL}_0$.

2. Установить управляющие переключатели в положения:

а) $\overline{WR}$ = логическая 1;

б) $IO/\overline{M}$ = логический 0. При этом система переводится в режим работы с памятью.

в) $\overline{RD}$ = логический 0. В результате система переводится в режим чтения памяти. При этом $\overline{BMEMR}$ принимает значение 0.

3. После шага 2(в) проверить наличие сигнала логического 0 на выводе 20 выбора блока IC₁₄ (выбирается данный блок).

4. Проверить наличие сигнала логической 1 на выводе 20 выбора блока IC₁₅. (Этот блок не выбирается.) При возникновении ошибки на шагах 3 или 4 можно статически проверить схемы дешифрирования при фиксировании системы в заданном режиме.

Процедура проверки правильности функционирования сигнала выбора блока на выводе 20 IC₁₅

5. Установить системный адрес $BA_0—BA_{15}$ в 0400. Из распределения памяти системы известно, что ПЗУ₁ соответствует адресное пространство 0400₁₆—07FF₁₆. В результате генерируется логический уровень 0 сигнала $\overline{SEL}_1$.

6. Система должна перейти в режим, в котором сигнал соответствует уровню логического 0.

7. Проверить наличие сигнала уровня логического 0 на выводе 20 IC₁₅.

8. Проверить наличие сигнала 1 на выводе 20 IC₁₄ (этот блок не выбран). При возникновении ошибки на шагах 7 или 8 можно статически проверить схемы дешифрирования при фиксировании системы в заданном режиме.

Теперь проверка правильности дешифрирования техническими средствами ПЗУ₀ и ПЗУ₁ на основе электрического состояния адресной шины $BA_0—BA_{15}$ и шины управления системы завершена.

Проверка подачи питания на ПЗУ

ППЗУ 2708 требует наличия трех источников питания: $+12\text{ В}$, $+5\text{ В}$ и -5 В . Проверим правильность электрического подсоединения этих трех источников к следующим выводам:

1. $+12\text{ В}$ — к выводу 19 (IC_{14} , IC_{15}).
2. $+5\text{ В}$ — к выводу 24 (IC_{14} , IC_{15}).
3. -5 В — к выводу 21 (IC_{14} , IC_{15}).

Указанные уровни напряжения должны существовать как для IC_{14} , так и для IC_{15} . Теперь проверка правильности электрического интерфейса всех устройств ПЗУ с системой завершена, и показано, что соответствующие технические средства функционируют.

10.9. Проверка интерфейса ОЗУ системы

При проверке интерфейса ОЗУ необходимо рассмотреть следующие пять основных типов связей:

- 1) интерфейс шины данных с ОЗУ;
- 2) интерфейс адресной шины с ОЗУ;
- 3) интерфейс схем выбора блока с ОЗУ;
- 4) интерфейс схем разрешения записи с ОЗУ;
- 5) подачу питания на ОЗУ.

Вспомним, что в случае ПЗУ осуществлялась проверка лишь четырех типов связей. В случае ОЗУ возникает дополнительная необходимость в проверке интерфейса схем РАЗРЕШЕНИЕ ЗАПИСИ. Данный тип интерфейса существует для ОЗУ, поскольку система способна осуществлять не только чтение данных из ОЗУ, что характерно и для ПЗУ, но также и запись данных в ОЗУ.

Проверка адресации ОЗУ

Проверку сигналов на входах ОЗУ будем проводить точно таким же образом, как это делалось в случае ПЗУ, т. е. путем выдачи адресного сигнала посредством УТСС и последующего контроля соответствующих адресных входов ОЗУ. ОЗУ системы состоит из блоков IC_{17} , IC_{18} , IC_{19} и IC_{20} . Поэтому при проверке адресных входов необходимо убедиться в правильности соединения каждой адресной линии со всеми четырьмя интегральными схемами. Повторять процедуру проверки адресных входов ОЗУ нет необходимости, поскольку это уже делалось достаточно подробно в отношении адресных входов ПЗУ системы.

Проверка шины данных ОЗУ

Проверка шины данных ОЗУ проводится опять так же, как это делалось в случае ПЗУ. Отличие состоит в том, что два блока ОЗУ включают ОЗУ 1 К $\times$ 8, поскольку используемые ОЗУ представляют собой устройства 2114, организованные как 1 К $\times$ 4 статическое ОЗУ. Для создания ОЗУ системы 1 К $\times$ 8 два устройства 2114 соединяются параллельно. Обращаясь снова к схеме системы (рис. 10.7) замечаем, что IC₁₇ и IC₁₈ составляют одно ОЗУ системы 1 К $\times$ 8, а IC₁₉ и IC₂₀ — другое ОЗУ 1 К $\times$ 8.

Линии BD₀—BD₃ шины данных соединяются с IC₁₇ и с IC₁₉, а линии BD₄—BD₇ — с IC₁₈ и с IC₂₀, как это показано на рис. 10.7.

Для проверки правильности соединения соответствующих линий шины данных с требуемыми блоками ОЗУ используем ту же процедуру, которая применялась при проверке шины данных ПЗУ. При этом система устанавливается в режим ЧТЕНИЕ с устройства ввода-вывода с подачей адреса 0000 на адресную шину (линии BA₀—BA₁₅). Далее поочередно заземляются линии шины данных ОЗУ BD₀—BD₇ и проверяется наличие логического уровня 0 сигнала на соответствующих светоизлучающих диодах УТСС. Отметим, что линии BD₀—BD₃ шины данных соединены с IC₁₇ и IC₁₉. Поэтому при проверке этих линий заземляется вывод 14 IC₁₉ и определяется выключение светоизлучающего диода D₀. Далее заземляется вывод 14 IC₁₇ и фиксируется выключение D₀. Данная процедура повторяется до тех пор, пока не будут проверены все линии BD₀—BD₇ шины данных на правильность их соединения с соответствующим блоком ОЗУ.

Проверка сигнала РАЗРЕШЕНИЕ ЗАПИСИ в ОЗУ

Перейдем к обсуждению техники проверки сигнала РАЗРЕШЕНИЕ ЗАПИСИ в ОЗУ системы. Отметим, что этот сигнал параллельно подается на вывод 10 IC₁₇ и IC₁₈, а также параллельно поступает на вывод 10 IC₁₉ и IC₂₀. Это вызвано тем, что каждое ОЗУ 1 К $\times$ 8 состоит из двух параллельно соединенных блоков ОЗУ меньшего объема. При подаче сигнала РАЗРЕШЕНИЕ ЗАПИСИ должны быть выполнены следующие условия. Система должна находиться в режиме записи данных в память, а на адресную шину должен быть подан адрес ОЗУ, в которое производится запись. Например, для успешной выдачи сигнала РАЗРЕШЕНИЕ ЗАПИСИ на блоки IC₁₇ и IC₁₈ на адресной шине должен содержаться адрес этих блоков, т. е. адресная шина должна содержать адрес 1000—13FF на адресных линиях BA₀—BA₁₅.

**Процедура проверки сигнала
РАЗРЕШЕНИЕ ЗАПИСИ
для блоков IC₁₇ и IC₁₈**

1. Установить адресную шину ($BA_0—BA_{15}$) в состояние 1000. При этом система переведет сигнал $\overline{SEL}_4$ в состояние логического 0. Сигнал $\overline{SEL}_4$ подается на выводы 1 и 4 IC₁₂.

2. Установить с помощью управляющих разрядов УТСС режим, при котором $IO/\overline{M}$ соответствует уровню логического 0. Это переводит систему в режим работы с памятью. Сигнал $\overline{RD}$ должен соответствовать уровню логической 1 (нерабочее состояние). Сигнал $\overline{WR}$ должен соответствовать уровню 0, что определяет наличие режима РАЗРЕШЕНИЕ ЗАПИСИ. При этом состоянии управляющих разрядов устанавливается сигнал $\overline{BMEMW}$, представляющий собой буферизованный сигнал ЗАПИСЬ в память системы. Этот сигнал поступает на вывод 1 IC₁₂.

Отметим, что в соответствии с электрическим состоянием адресной шины сигнал $\overline{SEL}_4$ находится в состоянии логического 0, а буферизованный сигнал ЗАПИСЬ в память — также в состоянии 0 в соответствии с логическим состоянием шины управления. При этих условиях на выводе 6 IC₁₂ поддерживается сигнал уровня логического 0. Заметим, что вывод 6 IC₁₂ также соединяется с выводом 10 IC₁₇ и IC₁₈, поскольку эти устройства, как упоминалось ранее, работают параллельно.

3. Проверить далее наличие уровня логического 0 на выводе 10 IC₁₇ и IC₁₈, как это должно быть.

4. Проверить наличие уровня логической 1 на выводе 10 IC₁₉ и IC₂₀. Это определяет наличие сигнала РАЗРЕШЕНИЕ ЗАПИСИ для IC₁₉ и IC₂₀. Если на шаге 3 или 4 обнаруживается ошибка, необходимо проверить информационный канал от УТСС микропроцессора 8085 к блоку ОЗУ.

5. Проверить подачу сигнала РАЗРЕШЕНИЕ ЗАПИСИ на блоки IC₁₉ и IC₂₀. По таблице распределения памяти видно, что адресное пространство IC₁₉ и IC₂₀ (ОЗУ₂) есть 1400—17FF. Установить адресные переключатели таким образом, чтобы на линии адресной шины $BA_0—BA_{15}$ был подан адрес 1400₁₆. При этом сигнал $\overline{SEL}_5$ с вывода 6 IC₁₂ принимает логический уровень 0. Сигнал $\overline{SEL}_5$ подается также на выводы 10 и 13 IC₁₂.

6. Теперь в системе продолжает поддерживаться сигнал $\overline{BMEMW}$. При логическом уровне 0 сигнала на выводах 13 и 12 IC₁₂ на выводе 11 IC₁₂ будет также сигнал логического уровня 0.

7. Проверить наличие логического уровня 0 на выводе 10 IC₁₉ и IC₂₀. Это рабочий режим для разрешения записи в ОЗУ₂.

8. Проверить наличие логического уровня 1 на выводе 10 IC₁₇ и IC₁₈. Это нерабочий режим для разрешения записи в ОЗУ₁. Если на шаге 7 или 8 обнаруживается ошибка, проверить снова информационный канал от ОЗУ к УТСС, чтобы локализовать неисправность в этом канале.

Таким образом, осуществлена проверка правильности прохождения сигнала ЗАПИСЬ в ОЗУ₁ и ОЗУ₂ системы. Проверим правильность прохождения сигнала по линии выбора блока ОЗУ₁ и ОЗУ₂. Эта линия должна находиться в активном состоянии (уровень 0) при чтении данных из ОЗУ или при их записи в ОЗУ.

Обсудим процедуру проверки линии выбора блока ОЗУ. Из рис. 10.7 видно, что устройство IC₁₃ является вентилем И типа 74LS08. Линия выбора блока соединяется с выходом одного из четырех вентилях, скомпонованных в устройстве 74LS08. IC₁₇ и IC₁₈ подсоединены к выводу 3 IC₁₃, а IC₁₉ и IC₂₀ — к выводу 6 IC₁₃. Отметим, что одним входом И вентилях является сигнал РАЗРЕШЕНИЕ ЗАПИСИ в ОЗУ₁ или ОЗУ₂, т. е. с выводом 2 IC₁₃ связан сигнал РАЗРЕШЕНИЕ ЗАПИСИ в ОЗУ₁, а с выводом 5 IC₁₃ — такой же сигнал ОЗУ₂.

Необходимо проверить, что при выдаче сигнала РАЗРЕШЕНИЕ ЗАПИСИ в ОЗУ₁ или ОЗУ₂ системы активизируется требуемая линия выбора блока ОЗУ. Этот факт позволяет осуществить проверку одного режима выбора блока точно таким же образом, как это делалось в случае проверки сигнала разрешения записи в ОЗУ₁ или ОЗУ₂. Вначале система устанавливается в соответствующий режим для проверки сигнала разрешения записи в ОЗУ₁, как это описывалось ранее, и определяется наличие логического уровня сигнала 0 соответственно выбору блока ОЗУ. Далее система устанавливается в режим подачи сигнала разрешения записи в ОЗУ₂, как описано выше. Затем проверяется наличие логического уровня 0 сигнала выбора блока ОЗУ₂.

Сигнал выбора блока ОЗУ₁ и ОЗУ₂ также должен иметь логический уровень 0 при чтении данных из ОЗУ₁ и ОЗУ₂. Это означает необходимость перевода системы в соответствующий режим, при котором выполнено условие ЧТЕНИЕ из ОЗУ₁ или ОЗУ₂. Для этого можно использовать следующую процедуру:

1. Установить системный адрес (BA₀—BA₁₅) 1000. В результате сигнал $\overline{SEL}_4$ (вывод 5 IC₁₁) имеет логический уровень 0. Этот сигнал осуществляет выбор ОЗУ₁.

2. Установить управляющие разряды УТСС в состояние, соответствующее ЧТЕНИЮ из памяти. Это осуществляется путем установки сигнала IO/ $\overline{M}$ на логический уровень 0. Сигнал RD принимает значение логического 0, а сигнал WR — логиче-

ской 1. При этих условиях сигнал $\overline{\text{ВМЕМР}}$ должен иметь уровень логического 0.

При одновременном существовании сигналов $\overline{\text{ВМЕМР}}$ и SEL_4 уровня логического 0 вывод 3 IC_{12} будет соответствовать также уровню логического 0, что является для системы условием чтения из ОЗУ.

3. Проверить наличие логического уровня 0 сигнала выбора блока на выводе 8 IC_{17} и IC_{18} . Отметим, что сигнал выбора блока принимает значение логического уровня 0, поскольку на вывод 1 IC_{13} подается уровень 0 вследствие наличия логического уровня 0 на выводе 3 IC_{12} .

4. Проверить наличие логического уровня 1 сигнала выбора блока ОЗУ₂ (нерабочий режим). Если на шаге 3 или 4 обнаруживается ошибка, необходимо локализовать неисправность в системе.

5. Проверить подачу сигнала выбора блока ОЗУ₂ при выполнении ЧТЕНИЯ из памяти для ОЗУ₂. С этой целью установить адрес 1400 на адресной шине системы ($\text{BA}_0\text{—BA}_{15}$) с помощью УТСС.

6. Система должна быть переведена в режим ЧТЕНИЕ из памяти. Отметим, что при нахождении системы в данном режиме и выдаче сигнала $\overline{\text{SEL}}_5$ (с помощью адресной шины) на выводе 8 IC_{12} поддерживается логический уровень 0. В результате на вывод 4 IC_{13} также подается логический уровень 0, что в свою очередь приводит к поддержанию логического уровня 0 на выводе 6 IC_{13} . Проверить наличие логического уровня 0 входного сигнала выбора блока ОЗУ₂.

7. Проверить наличие логического уровня 1 сигнала выбора блока ОЗУ₁ (нерабочее состояние). Если на шаге 6 или 7 обнаруживается ошибка, необходимо определить местоположение неисправности.

Допустим, что описанные выше проверки подтвердили правильность работы технических средств, что определяет правильность соединения адресной шины, а также шины данных с ОЗУ₁ и ОЗУ₂. Кроме того, шина управления системы осуществляет правильный выбор ОЗУ₁ и ОЗУ₂ при обращении к памяти для чтения или записи.

Наконец, осталось проверить правильность подачи питания на ОЗУ₁ и ОЗУ₂. При работе статического ОЗУ 2114 используется лишь источник +5 В, поэтому необходимо проверить наличие уровня напряжения +5 В на выводе 18 IC_{17} , IC_{18} , IC_{19} и IC_{20} . Показав, что источник питания функционирует удовлетворительно, мы тем самым завершаем проверку правильности интерфейса ПЗУ и ОЗУ с микропроцессорной системой на базе устройства 8085, имеющей архитектуру с 3 шинами.

10.10. Отладка интерфейса клавишного пульта

Рассмотрим отладку интерфейса клавишного пульта системы, поскольку мы используем тот же пульт, который был описан в гл. 4. Подробное обсуждение отладки этого интерфейса было проведено в гл. 5, поэтому повторение описанной там процедуры излишне. Если у читателя возникнут какие-либо вопросы в этом смысле, он сможет обратиться к гл. 5. В этой главе отладка интерфейса клавишного пульта обсуждалась для системы, создаваемой на базе микропроцессора Z80. В настоящей главе рассматривается система на основе микропроцессора 8085. Однако мы достаточно подробно изучили УТСС для микропроцессора 8085, чтобы провести такую же последовательность проверок, которая была описана в гл. 5 в отношении отладки клавишного пульта системы на базе Z80.

10.11. Отладка интерфейса устройства отображения

Подробное описание отладки интерфейса устройства отображения было приведено в гл. 5. В настоящей главе рассматривается точно такое же устройство отображения, что и в гл. 4. Поэтому, если у читателя имеются какие-либо неясности относительно отладки интерфейса данного устройства, он может обратиться к гл. 5, где соответствующие вопросы обсуждаются достаточно подробно. Изложение материала в гл. 5 ориентировано на микропроцессор Z80. Однако в гл. 10 даются обширные сведения по использованию УТСС для микропроцессора 8085, так что читатель в состоянии осуществить необходимые проверки для микропроцессора 8085 по аналогии с теми, которые описаны в гл. 5 для микропроцессора Z80.

10.12. Отладка схем подключения ППЗУ к системе

Рассмотрим теперь отладку схем ввода-вывода, предназначенных для подключения программируемого ППЗУ к системе. Эти схемы представлены на рис. 10.12. При обсуждении интерфейса с программируемым ППЗУ для рассматриваемой системы необходимо выделить пять основных функций технических средств системы, а именно:

1. Выдача адреса на программируемое ППЗУ.
2. Ввод и вывод данных ППЗУ;
3. Подача питания на ППЗУ.
4. Подача импульса программирования на ППЗУ.
5. Выдача импульса выбора блока/РАЗРЕШЕНИЕ ЗАПИСИ.

СИ.

Последние две функции описаны в отдельных разделах настоящей главы. Рассмотрим выдачу адреса на программируемое ППЗУ.

Из схемы рис. 10.12 видно, что выдача адреса на ППЗУ осуществляется посредством двух двоичных фиксаторов — IC₉ и IC₁₀. Устройство IC₉ есть 8-разрядный фиксатор типа 74LS374, а IC₁₀ — 4-разрядный фиксатор типа 74LS175. Отметим, что входом для 74LS374 являются линии BD₀—BD₇, а для 74LS175 — линии BD₀—BD₁. Это говорит о возможности пере-

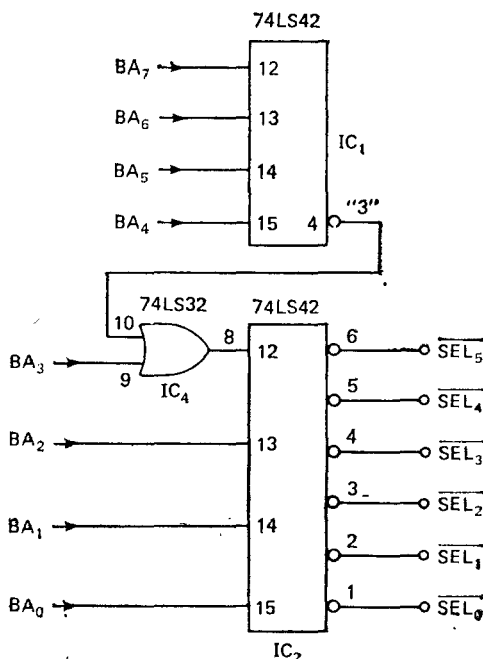


Рис. 10.13. Часть схемы рис. 10.12, генерирующая код выбора.

дачи 10-разрядного адреса ППЗУ, что согласуется с потребностями ППЗУ 1 К×8, для которого необходимо 10 адресных линий. Устройство 74LS374 IC₉ формирует младший байт адреса ППЗУ, а устройство 74LS175 — два старших разряда адреса.

Отметим, что синхронизация 8-разрядного фиксатора 74LS374 осуществляется с вывода 11 IC₃. Сигнал на этом выводе имеет логический уровень 0, если одновременно имеют уровень 0 сигналы на выводах 12 ($\overline{SEL_3}$) и 13 ($\overline{BIOW}$). Такие условия создаются, когда адресная шина переводит сигнал $\overline{SEL_3}$ к уровню логического 0, а шина управления генерирует буферированный сигнал записи при вводе-выводе ($\overline{BIOW}$). Рассмотрим генерирование сигнала $\overline{SEL_3}$ с помощью технических средств.

На рис. 10.13, который является частью рис. 10.12, показаны схемы дешифрирования выбора для всех выходных портов, приведенных на схеме рис. 10.12. Отметим, что адресные линии BA_4 и BA_7 подсоединяются к устройству 74LS42 IC_1 . Если состояние BA_4 — BA_7 соответствует значению 3, то сигнал на выводе 4 IC_1 соответствует логическому уровню 0. При этом на вывод 10 IC_4 подается уровень 0, что отпирает данный вентиль ИЛИ.

Линии младших разрядов адреса BA_0 — BA_2 подсоединены к устройству 74LS42 IC_2 , а линия BA_3 —к выводу 9 IC_4 . Если

Выход №1	IC ₂						Код выбора	Адрес
	2	3	4	5	6			
0	1	1	1	1	1	"0"	30 ₁₆	
1	0	1	1	1	1	"1"	31 ₁₆	
1	1	0	1	1	1	"2"	32 ₁₆	
1	1	1	0	1	1	"3"	33 ₁₆	
1	1	1	1	0	1	"4"	34 ₁₆	
1	1	1	1	1	0	"5"	35 ₁₆	

Рис. 10.14. Логические условия для адресной шины при установке требуемого кода выбора.

на BA_3 подается логический уровень 1, на вывод 8 IC_4 поступает сигнал логического уровня 0. При этом условии на вывод 12 IC_2 также поступает сигнал уровня 0. В результате открывается IC_2 .

Три линии младших разрядов адреса BA_0 — BA_2 теперь определяют, на какую из линий выбора $\overline{SEL}_0$ — $\overline{SEL}_5$ подается логический уровень 0. Это означает, что для адресной шины системы все сигналы $\overline{SEL}_0$ — $\overline{SEL}_5$ соответствуют значениям 30₁₆, 31₁₆, 32₁₆, 33₁₆, 34₁₆, 35₁₆. Состояния адресной шины и соответствующие коды выбора приведены на рис. 10.14. Код выбора для каждой линии выбора дан в кавычках.

Снова обращаясь к рис. 10.12, замечаем, что данные записываются в IC_9 , если сигнал $\overline{SEL}_3$ и буферизированный сигнал записи при вводе-выводе одновременно соответствуют уровню логического 0. Поэтому для проверки работы технических средств для данного тракта можно рекомендовать следующее (при необходимости можно обратиться к приведенным ранее процедурам):

1. Установить систему таким образом, чтобы сигнал $\overline{SEL}_3$ имел логический уровень 0 в зависимости от состояния линий адресной шины системы BA_0 — BA_7 .

2. Установить управляющие переключатели УТСС так, чтобы перевести систему в режим, при котором сигнал $\overline{BIOW}$ активен.

3. Установить шину управления в состояние, при котором сигнал $\overline{\text{BIOW}}$ неактивен.

4. При переводе сигнала $\overline{\text{BIOW}}$ в нерабочее состояние данные с шины данных записываются в 8-разрядный фиксатор IC_9 . Теперь, проверяя с помощью логического пробника уровни напряжения на выходах 1, 2, 3, 4, 5, 6, 7 и 8 панели программируемого ППЗУ, можно определить, что данные со входов IC_9 переданы на выходы IC_9 .

Затем необходимо проверить правильность функционирования устройства $74\text{LS175 } \text{IC}_{10}$ в условиях системы. С этой целью можно использовать следующую процедуру:

5. Перевести адресные линии $\text{BA}_0\text{—BA}_7$ в логическое состояние, при котором сигнал $\overline{\text{SEL}}_4$ имеет активный уровень 0.

6. Установить линии данных $\text{BD}_0\text{—BD}_1$ на входах устройства 74LS175 в состояние логической 1.

7. Установить систему в состояние, в котором является нерабочим буферизованный сигнал ЗАПИСЬ при вводе-выводе. В этом случае вывод 9 IC_{10} должен быть в состоянии логического 0.

8. Теперь буферизованный сигнал записи при вводе-выводе в систему не действует.

9. Проверить, что данные с выводов 4 и 5 устройства 74LS175 переданы на выходы 2 и 7 этого устройства. Это можно сделать путем проверки выводов 22 и 23 панели программируемого ППЗУ. При обнаружении ошибки на шаге 9 необходимо последовательно шаг за шагом проследить всю процедуру с целью локализации ошибки в аппаратуре.

Таким образом реализуется общая функция системы и проверяется правильность ее выполнения. При положительном результате проверки можно перейти к следующему шагу процедуры. В противном случае осуществляется возврат на шаг назад. Это позволит определить местоположение ошибки.

Теперь доказано, что адрес может быть передан в ППЗУ с помощью технических средств. Далее проверим возможность записи с помощью микропроцессора данных в ППЗУ и возможность чтения данных из ППЗУ посредством микропроцессора и технических средств системы.

Отметим, что в процессе программирования ППЗУ микропроцессор в некоторые промежутки времени осуществляет ввод данных из ППЗУ (чтение данных из ППЗУ). А в остальное время он выводит данные в ППЗУ (при генерации данных для программирования ППЗУ). Необходимо проверить, что в обоих случаях — при вводе и выводе данных ППЗУ — иххождение в системе обеспечивается электрически.

Вначале рассмотрим, каким образом осуществляется разрешение ввода в ППЗУ данных из микропроцессора. Данная

функция в основном реализуется посредством IC_5 рис. 10.12. Тракт ввода данных в ППЗУ с шины данных системы показан на рис. 10.15. Рисунок 10.15 является частью рис. 10.12. Отме-

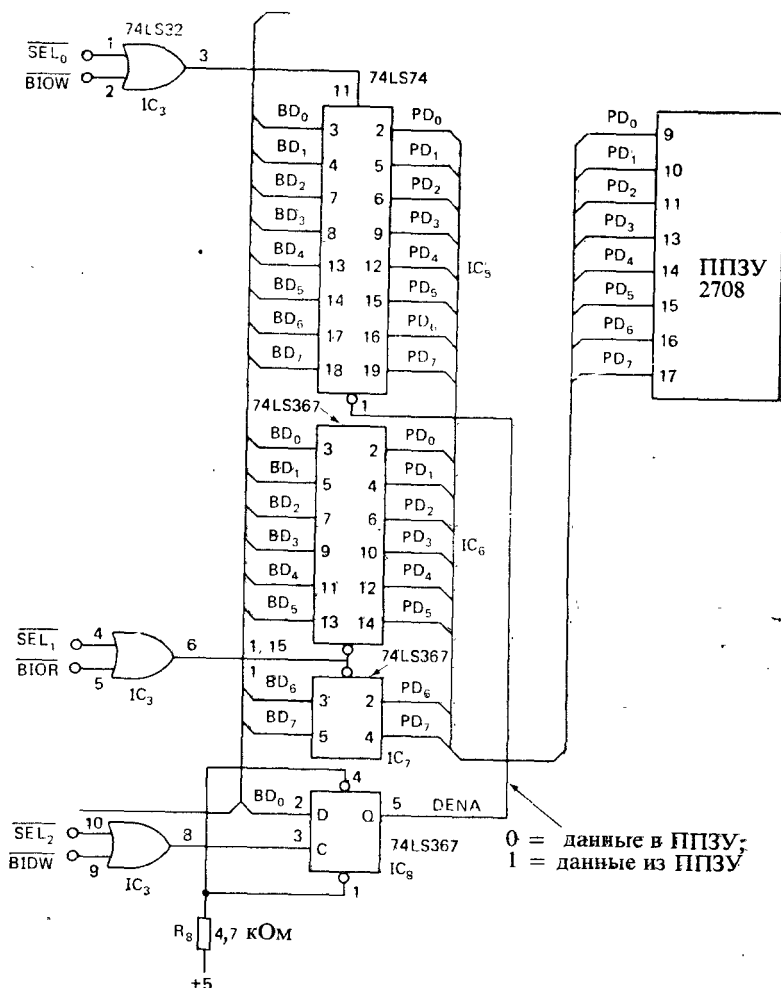


Рис. 10.15. Часть схемы рис. 10.12, определяющая тракт ввода-вывода данных для ППЗУ.

тим, что вывод 1 IC_5 подсоединен к выводу 5 фиксатора 74LS74. Если вывод 1 IC_5 находится в состоянии логического 0, то разрешен ввод данных с выхода IC_5 в ППЗУ. Ввод этих данных запрещен, если ППЗУ выдает данные в систему. Это предотвращает конфликты за обладание шиной данных между дан-

ными, выводимыми из ППЗУ, и данными, генерируемыми IC₅.

Прежде чем записывать данные в ППЗУ необходимо убедиться, что IC₅ открыт посредством сигнала с вывода 1. Это означает, что следует записать логический 0 в фиксатор 74LS74 (IC₈), что осуществляется с помощью разряда D₀ шины данных. Тактовые импульсы подаются на фиксатор с вывода 8 IC₃. Если сигнал SEL₂, а также буферизованный сигнал записи при вводе-выводе установлены и линия BD₀ шины данных имеет уровень 0, то уровень сигнала на выводе 1 IC₅ соответствует логическому 0.

Допустим, что на выводе 1 IC₅ поддерживается уровень 0. Отметим, что вход данных 8-разрядного фиксатора 74LS374 (IC₅) соединяется с шиной данных (BD₀—BD₇). Тем самым данные для программирования ППЗУ поступают на шину данных. После подачи данных на шину данных выполняется операция записи в IC₅. По этой операции данные передаются с шины данных на выход IC₅, т. е. осуществляется передача данных на программируемое ППЗУ.

Процедура проверки функционирования IC₅

1. Установить разряды адреса BA₀—BA₇ соответственно рабочему состоянию SEL₀.
2. Подать на линии BD₀—BD₇ данные, предназначенные для передачи на выход IC₅.
3. С помощью УТСС установить шину управления системы в состояние, в котором буферизованный сигнал записи при вводе-выводе принимает логический уровень 0.
4. Установить на линии управления состояние, соответствующее рабочему уровню сигнала BIOW. В этом случае данные с шины данных передаются на выходы IC₅.
5. Проверим, что данные на выходах IC₅ соответствуют исходным. В случае искажения данных на выходах IC₅ можно последовательно шаг за шагом с помощью УТСС пройти процедуру записи данных с целью локализации ошибки.

Рассмотрим теперь технику проверки возможности чтения микропроцессором данных из ППЗУ. Необходимость в этой операции возникает при установлении правильности записанных в ППЗУ данных. Для передачи данных из ППЗУ, в микропроцессор следует заблокировать выходы PD₀—PD₇ IC₅.

Вспомним, что для осуществления блокировки IC₅ необходимо подать уровень логической 1 на вывод 1. Это означает выполнение операции ЗАПИСИ при вводе-выводе в IC₈ при сигнале логической 1 на линии BD₀ шины данных, что приводит к подаче логической 1 на IC₅ и блокировке устройства 74LS374. В этом случае линии PD₀—PD₇ могут управляться лишь с выводов программируемого ППЗУ.

После выполнения операции записи в IC_8 и блокировки IC_5 можно реализовать следующие действия. Система устанавливается в режим чтения из IC_6 и IC_7 . Затем осуществляется поочередно заземление выходных линий ППЗУ и проверка наличия сигнала уровня логического 0 на соответствующих светоизлучающих диодах.

Процедура проверки уровня логического 0 на УТСС

1. Установить адресную шину системы в режим, при котором $\overline{SEL}_1$ в рабочем состоянии.

2. Установить шину управления системы так, чтобы буферизованный сигнал чтения при вводе-выводе имел рабочий уровень. При этом на вывод 6 IC_3 подается уровень логического 0, а устройства 74LS367 и IC_7 разблокируются. При данных условиях все светоизлучающие диоды УТСС загораются. В этом случае, конечно, действует предположение, что ППЗУ не подключено к панели программирования, в противном случае светоизлучающие диоды отображали бы данные, хранящиеся в устройстве.

3. Заземлить вывод 9 ППЗУ с помощью проводника, что соответствует логической 1. При этом разряд BD_0 линии данных принимает значение логического 0, что приводит к подаче уровня логического 0 на светоизлучающий диод D_0 .

4. Поочередно подать уровень логического 0 на выводы 10, 11, 13, 14, 15, 16 и 17 и проверить зажигание соответствующих светоизлучающих диодов УТСС. Таким путем осуществляется быстрая визуальная проверка электрического тракта от ППЗУ к микропроцессору 8085.

Теперь проверка правильности выдачи адреса на ППЗУ и функционирования шины данных завершена. Осталось определить правильность подачи питания на ППЗУ. Это может быть сделано путем контроля уровней напряжения на выводах 19, 24 и 21 с помощью вольтметра или осциллографа. Эти уровни должны быть равны соответственно 12 В, 5 В и — 5 В. Убедившись в правильности подачи таких уровней напряжения, перейдем к проверке следующего блока ППЗУ.

10.13. Проверка схем формирования импульса программирования

Проверим правильность функционирования схем, обеспечивающих уровни импульса программирования 26 В и 0 В. Рассматриваемые схемы показаны на рис. 10.16, который является частью рис. 10.12. В гл. 8 было описано функционирование схем рис. 10.16 с точки зрения прохождения электрических сигналов. Рассмотрим, каким образом микропроцессор управляет данными схемами. Из рис. 10.16 видно, что входы устройства

7406 IC_{12} являются выходами фиксатора IC_{11} . Он имеет всего четыре выхода. Выводы 5 и 9 (входы устройства 7406) связанные с выходами Q и $\bar{Q}$ одного разряда фиксатора 74LS175. Сигналы на этих выходах генерируются по сигналу входа D_2 фиксатора 74LS175. Последний соединяется с линией BD_1 шины данных. Это означает, что при передаче разряда BD_1 шины дан-

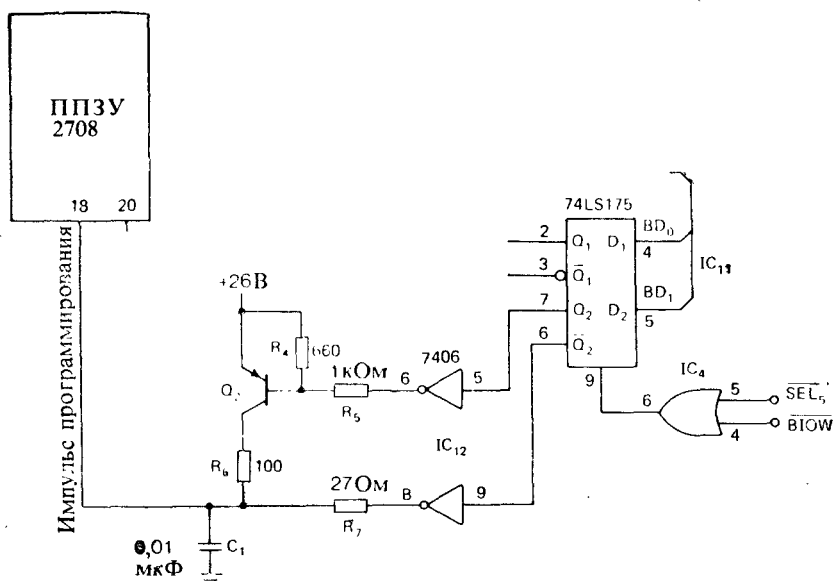


Рис. 10.16. Часть схемы рис. 10.12, определяющая технические средства управления импульсом программирования ППЗУ.

ных в фиксатор 74LS175 воздействие передается на выходы Q_2 и $\bar{Q}_2$.

Рассмотрим, каким образом данные с шины передаются в фиксатор IC_{11} . При выполнении системой операции выдачи кода выбора $\overline{SEL}_5$ данные с шины передаются в фиксатор IC_{11} . Выбор $\overline{SEL}_5$ осуществляется, если адресная шина ($BA_0—BA_7$) находится в логическом состоянии 35_{16} . Для проверки функционирования импульса программирования передается сигнал уровня логической 1 по линии шины данных BD_1 на выход Q_2 IC_{11} . При этих условиях импульс программирования ППЗУ на выводе 18 имеет уровень +26 В.

Процедура записи 1 на вход D_2 IC_{11}

1. Установить адресную шину системы ($BA_0—BA_7$) в состояние, в котором осуществляется подача сигнала $\overline{SEL}_5$.
2. Подать уровень 1 на линию BD_1 шины данных.

3. Установить управляющие разряды УТСС в состояние, в котором осуществляется подача сигнала B_{IO}W.

4. Установить шину управления системы таким образом, чтобы буферизированный сигнал записи B_{IO}W не подавался. В этом случае данные с линии BD₁ (логическая 1) вводятся в фиксатор 74LS175.

5. Импульс программирования на выводе 18 ППЗУ теперь имеет уровень +26 В. Если это не так, можно последовательно пройти по схеме и точно установить, почему импульс программирования не достигает указанного уровня при подаче 1 на вход D₂ IC₁₁.

6. Запись логического 0 на выход Q₂ IC₁₁ осуществляется точно так же, как и запись логической 1, за исключением того, что во время операции записи на соответствующей линии шины данных поддерживается уровень логического 0. После завершения операции записи импульс программирования на выводе 18 ППЗУ должен иметь уровень логического 0. Если на шагах 5 и 6 обнаружена ошибка, необходимо определить ее причину путем пошаговой процедуры статических проверок соответствующих схем.

10.14. Проверка функционирования схем ВЫБОРА БЛОКА и РАЗРЕШЕНИЯ ЗАПИСИ

Рассмотрим теперь схемы подачи уровня напряжения +12 В под управлением микропроцессора на линию ВЫБОРА БЛОКА РАЗРЕШЕНИЯ ЗАПИСИ (вывод 20 ППЗУ). На рис. 10.17, являющемся частью рис. 10.12, приведена схема управления импульсом ВЫБОРА БЛОКА РАЗРЕШЕНИЯ ЗАПИСИ, который подается на ППЗУ. Эта схема функционирует подобно схеме формирования импульса программирования, описанной ранее.

Основное различие между схемами ВЫБОРА БЛОКА РАЗРЕШЕНИЯ ЗАПИСИ и формирования импульса программирования заключается в том, что они управляются различными разрядами шины данных. Схемы ВЫБОРА БЛОКА РАЗРЕШЕНИЯ ЗАПИСИ управляются разрядом BD₀ шины данных. Таким образом, при записи 1 на выход Q₁ устройства 74LS175 IC₁ сигнал выбора блока разрешения записи должен достичь уровня +12 В с помощью транзистора T₁. При записи 0 на выход Q₁ уровень напряжения данного сигнала становится 0 В. Функционирование схем выбора блока разрешения записи может быть проверено путем выполнения операции ЗАПИСИ при установке логической 1 в разряде BD₀ шины данных. При этом на входе линии выбора блока разрешения записи

устанавливается уровень напряжения $+12$ В. Далее выполняется операция записи при установке логического 0 в разряде BD_0 шины данных. В результате на рассматриваемую линию подается уровень напряжения 0 В. Если при выполнении какой-либо из указанных операций на выходе транзистора T_1 получается неверный сигнал, то, прежде чем двигаться дальше, необходимо локализовать и устранить неисправность в схеме.

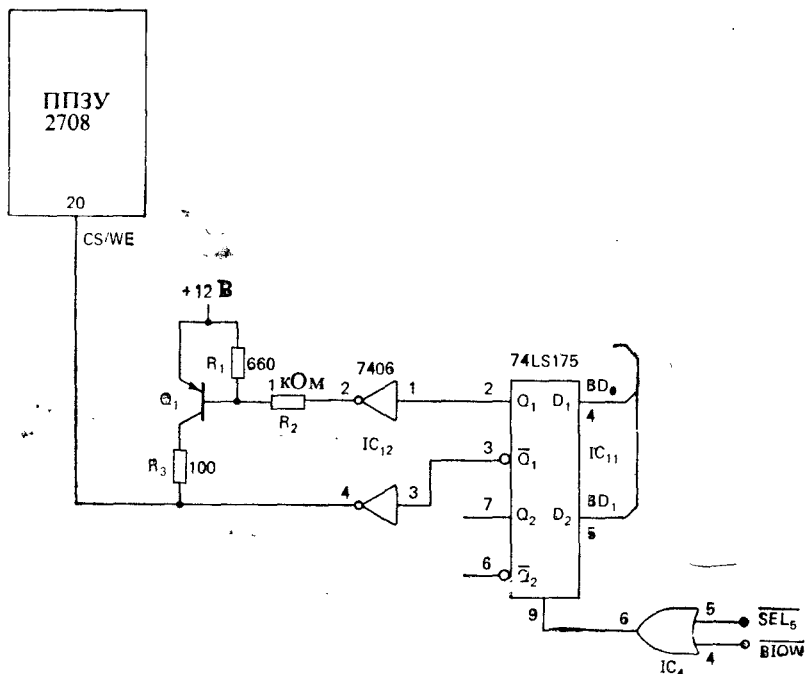


Рис. 10.17. Часть схемы рис. 10.12, определяющая технические средства управления импульсом $\overline{CS/WE}$ ППЗУ.

По мере углубления в детали технических средств строгие процедуры проверки системы, описанные в начале главы, упрощаются. Это вызвано тем, что наши знания о сущности перевода системы в режим записи или режим чтения посредством УТСС расширяются. При упоминании о переводе системы в какой-либо частный режим предполагаем, что читатель знает, каким образом этот режим реализуется посредством УТСС. Если же это не так, необходимо ознакомиться с ранее описанными процедурами перевода системы в режимы записи и чтения при операциях с памятью или с устройствами ввода-вывода посредством УТСС.

По завершении проверки технических средств формирования импульса программирования и выбора блока разрешения записи устанавливается факт правильной работы схем ввода-вывода ППЗУ в рамках всей системы. Проведение полного объема тестирования с использованием лишь УТСС и устройства контроля уровня напряжения является большим достижением. При изложении материала основной упор делается на точное понимание особенностей протекания коммуникационных процессов в условиях использования конкретного микропроцессора, т. е. для того чтобы эффективно использовать УТСС при настройке технических средств, необходимо знать, каким образом микропроцессор 8085 или любой другой выполняет с устройствами ввода-вывода или с памятью операции чтения и записи.

Убедившись в правильности функционирования технических средств системы, мы можем подключить микропроцессор и задать системе рабочий режим. При этом мы уверены, что и теперь система будет работать корректно. Подчеркнем еще раз, что УТСС не проверяет ошибки тактирования. Однако личный опыт автора позволяет утверждать, что если проверка с помощью УТСС выявила правильную работу технических средств, то и при нормальном быстродействии системы с установленным микропроцессором с большой вероятностью это оборудование будет функционировать безошибочно. Последнее доказано на работе с микропроцессорами 8080, 8085, Z80 и 6800 системы, описанной в гл. 8. Причем если возникает намерение использовать УТСС для настройки оборудования на базе микропроцессора 6800, то необходимо создать специальное устройство тестирования статических сигналов, схема которого приведена в приложении.

Затем, используя это устройство, реализуется полная процедура проверки, описанная в данной главе. Причем индикация операции ЧТЕНИЯ из памяти должна быть совместимой с соответствующей операцией микропроцессора 6800. При индикации операций ЗАПИСИ и ЧТЕНИЯ с устройствами ввода-вывода или операции ЗАПИСИ в память УТСС переводится в тот же режим, в котором эти операции выполняет микропроцессор 6800. В этом заключаются различия между УТСС, используемыми для различных микропроцессоров. Используемое тестирующее устройство должно быть совместимым с микропроцессором, который оно заменяет в системе, а также оно должно эмулировать этот микропроцессор. Это обобщенное свойство УТСС, позволяющее модифицировать существующее УТСС для проверки технических средств любой базовой микропроцессорной системы вне зависимости от того, какой из микропроцессоров 8080, 8085, Z80, 6800 (либо какой-нибудь другой, менее популярный) используется.

10.15. Выводы

В книге была описана последовательность проектирования технических средств и создания программного обеспечения системы, в которой одни и те же функции могут быть реализованы с помощью четырех различных микропроцессоров. Их выбор определялся тем, что присущие им общие архитектурные особенности широко используются в устройствах, применяемых на практике.

Также была изложена техника построения интерфейса устройств ввода-вывода и памяти с этими микропроцессорами (при общей или специальной организации ввода-вывода). Устройства ввода-вывода, на примере которых демонстрировалось построение интерфейса с системой — клавишный пульт и устройство отображения, — относились к числу широко распространенных, а выбор их основывался на стремлении проиллюстрировать общую концепцию построения интерфейса ввода-вывода, отвлекаясь от специфических свойств отдельных периферийных устройств, которые могут использоваться лишь с определенным микропроцессором, определенной микропроцессорной системой либо в ограниченной области применения.

Использование специальных периферийных устройств с данным микропроцессором способствует снижению числа выводов системы. Необходимо знать периферийные устройства, которые могут работать с микропроцессором, используемым в системе. Однако с изучением концепции построения интерфейса устройств ввода-вывода облегчается понимание технических данных и спецификаций отдельных периферийных устройств. Это объясняется тем, что по мере ознакомления с используемым периферийным устройством осмысливается значение его конкретных характеристик.

Была изложена техника проверки, или отладки технических средств системы, именуемая тестированием посредством статических сигналов. Существуют и другие технические приемы проверки технических средств системы, однако тестирование посредством статических сигналов отличается крайней простотой, поскольку опирается на знание инженерно-техническим персоналом особенностей взаимосвязи микропроцессора с памятью и устройствами ввода-вывода. Такая информация необходима при проектировании использования устройств ввода-вывода и памяти в микропроцессорной системе. Кроме того, использование устройства тестирования статических сигналов не связано с приобретением каких-либо дополнительных знаний.

Остается надеяться, что ознакомление с настоящей книгой облегчит для читателя использование, отладку и проектирование систем на базе 8-разрядных микропроцессоров.

Приложение

(2-19)

Назначение выводов корпуса микропроцессора INTEL 8080A

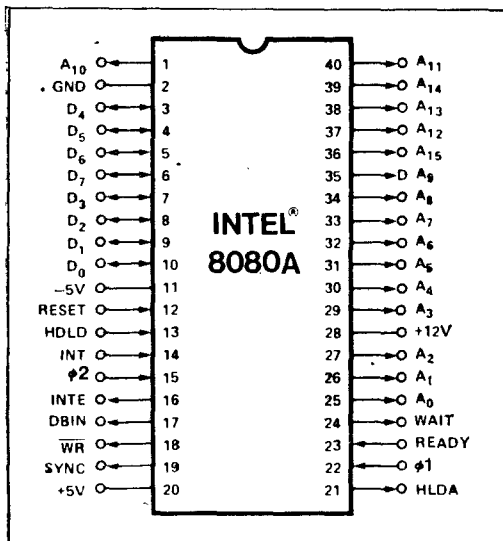


Рис. П.1. Назначение выводов микропроцессора Intel 8080A.

A₁₅—A₀ (выходы с тремя устойчивыми состояниями)

Выходы A₁₅—A₀ подключаются к шине данных, по которым адрес подается в память (допускается использование памяти объемом в 64К 8-разрядных слов) или на устройство ввода-вывода. Может быть использовано до 256 устройств ввода и до 256 устройств вывода. A₀ — младший разряд адреса.

D₇—D₀ (входы-выходы с тремя устойчивыми состояниями)

К выводам D₇—D₀ подключается шина данных, по которой обеспечивается двунаправленная передача команд и данных между ЦП, памятью и устройствами ввода-вывода. Кроме того, в начале каждого машинного цикла микропроцессор 8080A выводит на шину данных слово состояния. D₀ — младший разряд слова, передаваемого по шине данных.

SYNC (выход)

Синхронизирующий сигнал SYNC отмечает начало каждого машинного цикла.

DBIN (выход)

Сигнал DBIN информирует внешние схемы о том, что шина данных работает в режиме ввода. Этот сигнал должен использоваться для разрешения

ввода данных в микропроцессор 8080A из памяти или устройств ввода-вывода.

READY (вход)

Сигнал READY указывает микропроцессору 8080A, что на шину данных поступили данные из памяти или от внешних устройств. Этот сигнал используется для синхронизации ЦП при работе его с медленнодействующими памятью или устройствами ввода-вывода. Микропроцессор 8080A после выдачи адреса будет находиться в состоянии ожидания (этому состоянию соответствует низкий уровень сигнала на линии READY) до тех пор, пока не поступит сигнал READY. Кроме того, этот вход можно использовать для организации пошагового режима работы микропроцессора.

WAIT (выход)

Сигнал WAIT является сигналом подтверждения того, что ЦП находится в состоянии ожидания.

$\overline{WR}$ (выход)

Сигнал $\overline{WR}$ используется для управления при выполнении операции записи в память или при выводе данных на внешние устройства. До тех пор пока сигнал $\overline{WR}$ имеет активно низкий уровень ($\overline{WR}=0$), данные сохраняются на шине данных.

HOLD (вход)

Сигнал HOLD является сигналом запроса на перевод ЦП в состояние «Блокировка». В состоянии «Блокировка» внешнее устройство получает возможность управлять адресной шиной и шиной данных. Переход в состояние «Блокировка» осуществляется после окончания текущего машинного цикла при выполнении следующих условий:

- ЦП находится в состоянии HALT.

- ЦП находится в состоянии T2 или TW и сигнал READY активизирован. При переходе в состояние «Блокировка» адресная шина ($A_{15}-A_0$) и шина данных (D_7-D_0) будут переведены в состояние высокого сопротивления. ЦП выдает на вывод HLDA сигнал подтверждения перехода в состояние «Блокировка».

HLDA (выход)

Сигнал HLDA появляется в ответ на сигнал HOLD и указывает, что адресная шина и шина данных перешли в состояние высокого сопротивления. Сигнал HLDA вырабатывается при следующих условиях:

- ЦП находится в состоянии T3 и производится ввод данных из памяти или из устройства ввода.

- Наступил период в последовательности тактовых импульсов, следующий за состоянием T3 при выполнении операции записи в память или операции вывода на внешние устройства. В любом случае сигнал HLDA появляется после прохождения переднего фронта импульса #1, а переход в состояние высокого сопротивления происходит после прохождения переднего фронта импульса #2.

INTE (выход)

Состояние этого выхода определяется содержимым триггера разрешения внешнего прерывания. Состояние этого триггера может определяться посредством команды «Разрешение прерывания» и команды «Запрещение прерывания». Когда этот триггер «сброшен», ЦП не реагирует на прерывания. Он автоматически сбрасывается (последующие прерывания запрещаются) в состоянии T1 цикла команды, и, кроме того, он сбрасывается при поступлении сигнала RESET.

INT (вход)

Сигнал запроса на прерывание воспринимается ЦП либо в конце цикла выполнения текущей команды, либо в состоянии HOLD. Если ЦП находится

Таблица 1

Характеристики по постоянному току

$T_A = 0-70^\circ\text{C}$, $V_{DD} = +12 \text{ В} \pm 5\%$, $V_{CC} = +5 \text{ В} \pm 5\%$, $V_{BB} = -5 \text{ В} \pm 5\%$, $V_{SS} = 0 \text{ В}$,
если не оговорено противное

Обозначение	Параметр	Значение		Единица измерения	Условия испытаний
		минимальное	среднее максимальное		
V_{ILC}	Низкий уровень тактового напряжения	$V_{SS} - 1$	$V_{SS} + 0,8$	В	
V_{INC}	Высокий уровень тактового напряжения	9,0	$V_{DD} + 1$	В	
V_{IL}	Низкий уровень входного напряжения	$V_{SS} - 1$	$V_{SS} + 0,8$	В	
V_{IN}	Высокий уровень входного напряжения	3,3	$V_{CC} + 1$	В	
V_{OL}	Низкий уровень выходного напряжения		0,45	В	$I_{OL} = 1,9 \text{ мА}$ на всех выходах,
V_{OH}	Высокий уровень выходного напряжения	3,7		В	$I_{OH} = -150 \text{ мкА}$
I_{DD}	Средний ток при V_{DD}		40	мА	Режим $T_{CY} = 0,48 \text{ мс}$
I_{CC}	$\rightarrow \rightarrow V_{CC}$		60	мА	
I_{BB}	$\rightarrow \rightarrow V_{BB}$		0,01	мА	
I_{IL}	Ток утечки на входе		± 10	мкА	$V_{SS} \leq V_{IN} \leq V_{CC}$
I_{CL}	Ток ут. на вх. синхронизации		± 10	мкА	$V_{SS} \leq V_{TAKT} \leq V_{DD}$
I_{DL}	Ток ут. на шине данных в режиме ввода		-100	мкА	$V_{SS} \leq V_{IN} \leq V_{SS} + 0,8 \text{ В}$
I_{FL}	Ток ут. на адресной шине и шине данных в режиме HOLD		$-2,0$	мА	$V_{SS} + 0,8 \text{ В} \leq V_{IN} \leq V_{CC}$
			$+10$	мкА	$V_{адрес/данные} = V_{CC}$
			-100	мкА	$V_{адрес/данные} = V_{SS} + 0,45 \text{ В}$

в состоянии «Блокировка» или триггер разрешения прерывания сброшен, он не будет реагировать на прерывания.

RESET (вход)

Когда сигнал RESET активизируется, счетчик команд очищается. После подачи сигнала RESET ЦП начнет выполнение программы с команды, находящейся в памяти в ячейке с адресом 0. Кроме того, будут сброшены триггеры, определяющие состояния выходов INTE и HLDA. Отметим, что очистка флажков, аккумулятора, указателя стека и регистров не производится. $V_{SS}=0$ В, $V_{DD}=+12\pm 5\%$ В, $V_{CC}=+5\pm 5\%$ В, $V_{BB}=-5\pm 5\%$ В. $\phi 1, \phi 2$ — первая и вторая фазы последовательности тактовых импульсов. Эти входы не совместимы с TTL-схемами.

Допустимые предельные значения [8080A¹⁾]

Температура окружающей среды	0 — +70 °C
Температура устройства памяти	-65 — +150 °C
Все входные и выходные напряжения относительно напряжения V_{BB}	-0,3 — +20 В
V_{CC}, V_{DD} и V_{SS} относительно V_{BB}	-0,3 — +20 В
Мощность рассеяния	1,5 Вт

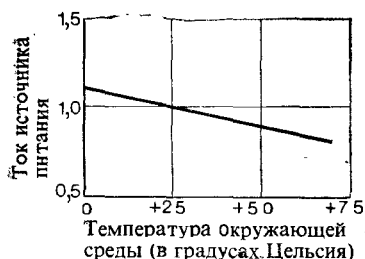


Рис. П.2. Зависимость тока источника питания от температуры окружающей среды. Зависимость нормализована: $\Delta I/\Delta T_A = -0,45\%/^{\circ}\text{C}$.

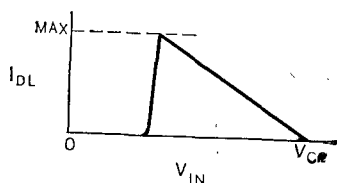


Рис. П.3. Зависимость тока утечки шины данных от уровня сигнала DBIN

Таблица 2

Емкостные характеристики

$T_A = 25^{\circ}\text{C}$, $V_{CC} = V_{DD} = V_{SS} = 0$ В, $V_{BB} = -5$ В

Обозначение	Параметр	Номинальное значение	Максимальное значение	Единица измерения	Условие испытания
C	Емкость на входах ϕ	17	25	пФ	$f_C = 1\text{ МГц}$
C _{NI}	Входная емкость	6	10	пФ	
C _{OUT}	Выходная емкость	10	20	пФ	

¹⁾ Функционирование устройства при значениях параметров, превышающих предельные, не допускается. Работа устройства при предельных значениях указанных параметров в течение длительного времени может изменить характеристики надежности устройства.

Примечание: Измерение временных характеристик выполняется при следующих напряжениях: на входах тактовых импульсов — уровень логич. «1» = 8,0 В, уровень логич. «0» = 1,0 В; на остальных входах — уровень логич. «1» = 3,3 В, уровень логич. «0» = 0,8 В; на выходах уровень логич. «1» = 2,0 В, уровень логич. «0» = 0,8 В.

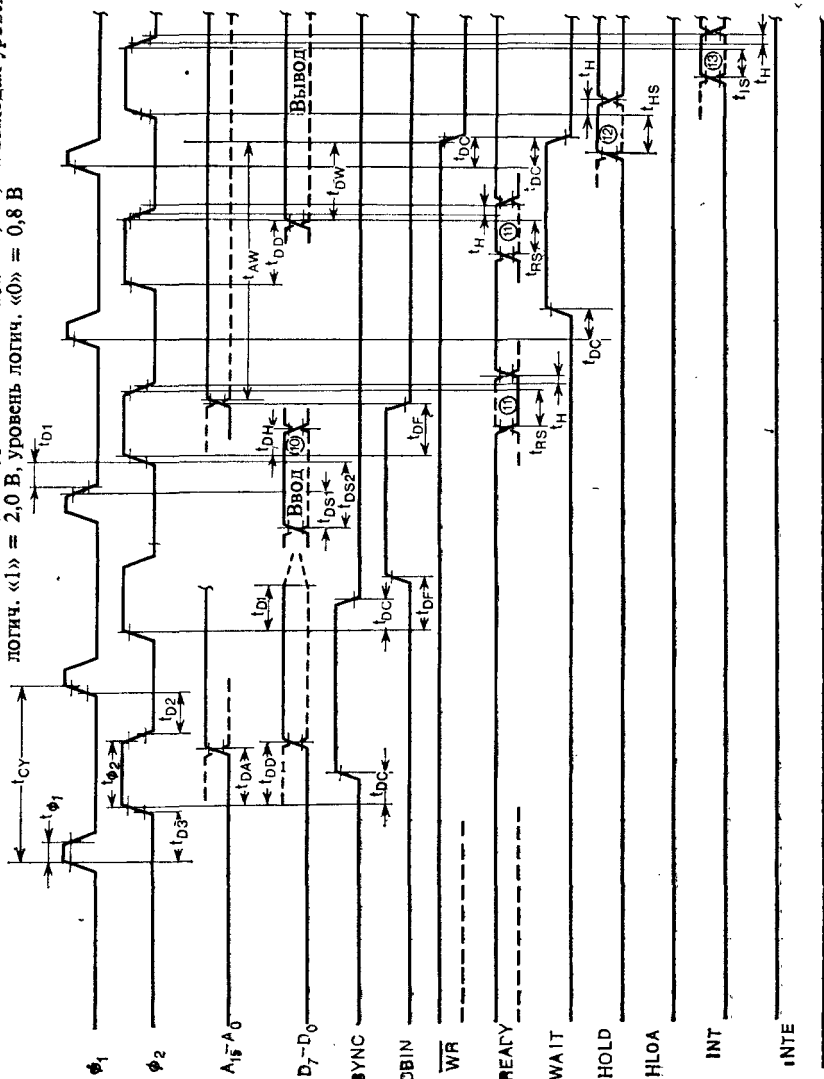


Рис. П.4. Временные диа-
граммы (8080A).

Таблица 3

Характеристики по переменному току (микропроцессор 8080 А)
 (Если не оговорено особо, то $T_A = 0 \text{—} 70^\circ\text{C}$, $V_{DD} = +12 \text{ В} \pm 5\%$,
 $V_{CC} = +5 \text{ В} \pm 5\%$, $V_{BB} = -5 \text{ В} \pm 5\%$, $V_{SS} = 0 \text{ В}$.)

Обозначение	Параметр	Значение						Единица измерения	Условие проверки
		мин.	макс.	мин. 0,1	макс. 0,1	мин. 0,2	макс. 0,2		
t_{CY}	Период следования тактовых импульсов	0,48	2,0	0,32	2,0	0,38	2,0	мкс	
t_r, t_f	Длительность переднего и заднего фронта тактовых импульсов $\phi 1$ и $\phi 2$	0	50	0	25	0	50	нс	
$t_{\phi 1}$	Ширина импульса $\phi 1$	60		50		60		нс	
$t_{\phi 2}$	Ширина импульса $\phi 2$	220		145		175		нс	
t_{D1}	Задержка импульса $\phi 2$ по отношению к заднему фронту импульса $\phi 1$	0		0		0		нс	
t_{D2}	Задержка импульса $\phi 1$ по отношению к заднему фронту $\phi 2$	70		60		70		нс	
t_{D3}	Время задержки импульса $\phi 2$ по отношению к импульсу $\phi 1$	80		60		70		нс	
t_{DA}	Время задержки подачи адреса относительно импульса $\phi 2$		200		150		175	нс	} $C_L = 100 \text{ пФ}$
t_{DD}	Время задержки выдачи данных относительно импульса $\phi 2$		220		180		200	нс	
t_{DC}	Задержка подачи сигналов SYNC, WR, WAIT, HLDA по отношению к переднему фронту импульсов $\phi 1$ и $\phi 2$		120		110		120	нс	} $C_L = 50 \text{ пФ}$
t_{DF}	Время задержки подачи сигнала DBIN относительно импульса $\phi 2$	25	140	25	130	25	140	нс	
t_{DI}	Время задержки перехода шины данных в режим ввода по отношению к переднему фронту импульса $\phi 2$		t_{DF}		t_{DF}		t_{DF}	нс	
t_{DSI}	Время установки данных на шине данных	30		10		20		нс	

Функциональное описание 8080A/8085A-2

Микропроцессор 8085A представляет собой 8-разрядный ЦП. Для него требуется источник питания с напряжением +5 В. Тактовая частота генератора тактовых импульсов для микропроцессора 8085A равна 3 МГц, а для микропроцессора 8085A-2 — 5 МГц. Таким образом, эти микропроцессоры обладают более высоким быстродействием по сравнению с микропроцессором 8080A. Кроме того, для построения систем, обладающих минимальными возможностями, предусмотрены следующие устройства: ОЗУ/IO — 8156, ПЗУ/IO — 8355 или ППЗУ/IO — 8755A.

Микропроцессор 8085A имеет двенадцать 8-разрядных регистров. Четыре из них, разбитые на пары, используются как два 16-разрядных регистра. Шесть других регистров могут использоваться либо как 8-разрядные, либо для образования 16-разрядных регистров. Регистры микропроцессора 8085A имеют следующее назначение:

Обозначение	Регистр	Содержимое
ACC или A	Аккумулятор	8 разрядов
PC	Счетчик команд	16-разрядный адрес
BC, DE, HL	Регистры общего назначения; HL — указатель данных	Шесть 8-разрядных, три 16 разрядных
SP	Указатель стека	16-разрядный адрес
F	Регистр флажков	5 флажков (8 разрядов)

В микропроцессоре 8085A использована мультиплексированная шина данных. Адрес передается по двум шинам: старший байт адреса — по шине адреса, а младший байт — по шине данных. В начале каждого машинного цикла младший байт адреса поступает на шину данных. Этот младший байт адреса может быть зафиксирован в любом 8-разрядном фиксаторе посредством подачи сигнала отпирания фиксатора адреса (ALE). В остальное время машинного цикла шина данных используется для передачи данных между микропроцессором и памятью или устройствами ввода-вывода.

Микропроцессор 8085A вырабатывает для шины управления сигналы $\overline{RD}$, $\overline{WR}$, S_0 , S_1 и $IO/\overline{M}$. Кроме того, он же выдает сигнал подтверждения прерывания $\overline{INTA}$. Сигнал $\overline{HOLD}$ и все прерывания синхронизируются с помощью внутреннего генератора тактовых сигналов. Для обеспечения простого последовательного интерфейса в микропроцессоре 8085A предусмотрены линия последовательного ввода данных (SID) и линия последовательного вывода данных (SOD).

Микропроцессор 8085A имеет также три вывода для маскируемых сигналов прерывания по вектору и один вывод для немаскируемого прерывания TRAP.

Прерывания и последовательный ввод-вывод

Микропроцессор 8085A имеет всего 5 входов для подачи сигналов прерываний: $\overline{INTR}$, RST 5.5, RST 6.5, RST 7.5 и TRAP. Сигнал $\overline{INTR}$ имеет такое же назначение, как и сигнал $\overline{INT}$ в микропроцессоре 8080A. Каждый из входов RST 5.5, RST 6.5 и RST 7.5 может программно маскироваться. Прерывание по входу TRAP не может быть маскировано. Если маска прерываний не установлена, то на указанные маскируемые прерывания микропроцессор будет реагировать, помещая при этом содержимое счетчика команд в стек и переходя к выполнению программы, адрес которой определяется вектором рестарта. Так как прерывание TRAP не может быть маскировано, при появле-

нии запроса прерывания на этом входе микропроцессор будет всегда переходить к выполнению программы, указанной вектором рестарта.)

Среди маскируемых прерываний есть прерывания двух различных типов. Входы сигналов прерываний RST 5.5 и RST 6.5 являются подобно входам INTR и входу INT микропроцессора 8080 чувствительными к уровню сигнала, а время распознавания соответствующих сигналов такое же, как и для сигнала INTR. Вход RST 7.5 является чувствительным к переднему фронту сигнала. Чтобы установить триггер, генерирующий внутренний запрос на прерывание, достаточно подать импульс на вход RST 7.5. Триггер запроса прерывания RST 7.5 остается установленным до тех пор, пока прерывание не будет обработано. После завершения обработки прерывания он автоматически сбрасывается.

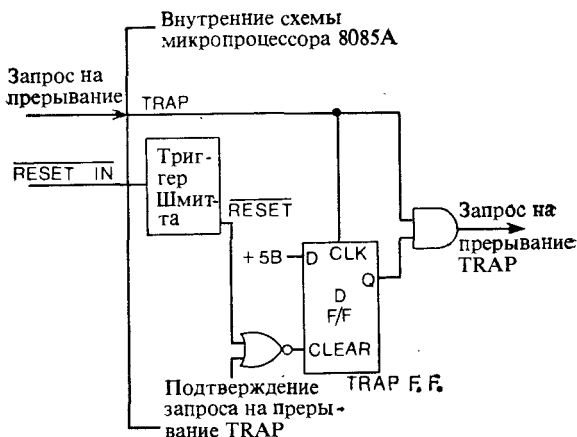


Рис. П.5. Схема приема сигналов TRAP и RESET IN.

сывается. Этот триггер также может быть сброшен с помощью команды SIM или посредством подачи сигнала RESET IN на соответствующий вход микропроцессора 8085A. Внутренний триггер запроса прерывания RST 7.5 может быть установлен подачей импульса на вывод RST 7.5 даже тогда, когда прерывание RST 7.5 маскировано. Состояние масок прерываний типа RST может быть изменено только командой SIM и сигналом RESET IN (см. описание команды SIM в гл. 6).

Каждому прерыванию приписан некоторый постоянный приоритет: сигнал TRAP имеет наивысший приоритет, затем идут сигналы RST 7.5, RST 6.5, RST 5.5, сигнал INTR имеет низший приоритет. Если на прерывание поступает более чем один запрос, то сначала обрабатывается прерывание, имеющее наивысший приоритет. Эта приоритетная схема не влияет на уже стартовавшие программы обработки прерываний. Но если прерывания разрешены до окончания выполнения программы обработки прерывания, то при поступлении запроса на прерывание по входу RST 5.5 программа обработки прерывания RST 7.5 прекратит работу.

Сигнал прерывания TRAP используется при возникновении катастрофических событий, например при повреждении источника питания или при возникновении неисправностей в пинах системы. Сигнал TRAP распознается системой подобно другим прерываниям, однако он имеет наивысший приоритет. Это прерывание не может быть маскировано. Вход TRAP является чувствительным и по фронту сигнала, и по его уровню. Сигнал TRAP должен выра-

сти до определенного уровня и оставаться на этом высоком уровне вплоть до выработки сигнала подтверждения прерывания. Следующее прерывание на входе TRAP может быть зарегистрировано только тогда, когда уровень сигнала на нем изменится от низкого до высокого уровня. Таким образом устраняются ложные срабатывания, которые могли бы возникнуть из-за шумов или других помех. На рис. П.5 представлена схема формирования запроса на прерывание TRAP, содержащаяся в микропроцессоре 8085A. Отметим, что во время обработки прерываний TRAP, RST 7.5, RST 6.5, RST 5.5, INTR, до тех пор пока не будет выполнена команда EI, запрещены все прерывания, кроме прерывания по входу TRAP. Прерывание TRAP является особым в том смысле, что оно запрещает прерывания, но не изменяет предыдущее состояние входа разрешения прерывания. Выполнение команды RIM после прерывания TRAP позволяет определить состояние маски прерываний независимо от того были ли разрешены или запрещены прерывания до наступления прерывания TRAP. Все последующие выполнения команды RIM дают текущее состояние маски прерываний. Выполнение команды RIM после прерываний по входам INTR, RST 5.5, RST 6.5 и RST 7.5 также будет информировать о текущем состоянии маски прерываний, обнаруживая, что прерывания заблокированы.

Допустимые предельные значения (8085A/8085A-2)

Температура окружающей среды	0—70 °C
Температура устройства памяти	—65—+150 °C
Напряжения на всех выводах по отношению к корпусу	—0,5—+7 В
Мощность рассеяния	1,5 Вт

Таблица 4

Характеристики по постоянному току
(Если особо не оговаривается, то $T_A=0-70^\circ\text{C}$, $V_{CC}=5\text{ В}\pm 5\%$, $V_{SS}=0\text{ В}$.)

Обозначение	Параметр	Значение		Единица измерения	Условия проверки
		мин.	макс.		
V_{IL}	Низкое входное напряжение	—0,5	+0,8	В	$I_{OL}=2\text{ мА}$ $I_{OH}=-400\text{ мкА}$ $V_{in}=V_{CC}$ $0,45\text{ В}\leq V_{out}\leq V_{CC}$
V_{IH}	Высокое входное напряжение	2,0	$V_{CC}+0,5$	В	
V_{OL}	Низкое выходное напряжение		0,45	В	
V_{OH}	Высокое выходное напряжение	2,4		В	
I_{CC}	Ток источника питания		170	мА	
I_{IL}	Ток утечки на входе		± 10	мкА	
I_{LO}	Ток утечки на выходе		± 10	мкА	
V_{ILR}	Низкий уровень на входе RESET	—0,5	+0,8	В	
V_{IHR}	Высокий уровень на входе	2,4	$V_{CC}+0,5$	В	

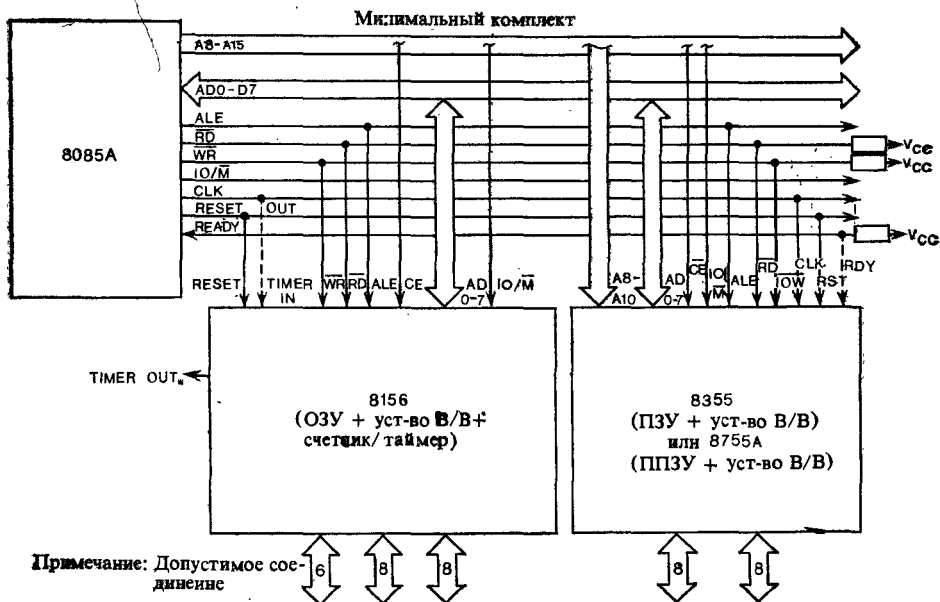


Рис. П.6. Микропроцессорная система MCS-85™. (Минимальный комплект. Связь с устройствами ввода-вывода реализована по аналогии с обращением к памяти.)

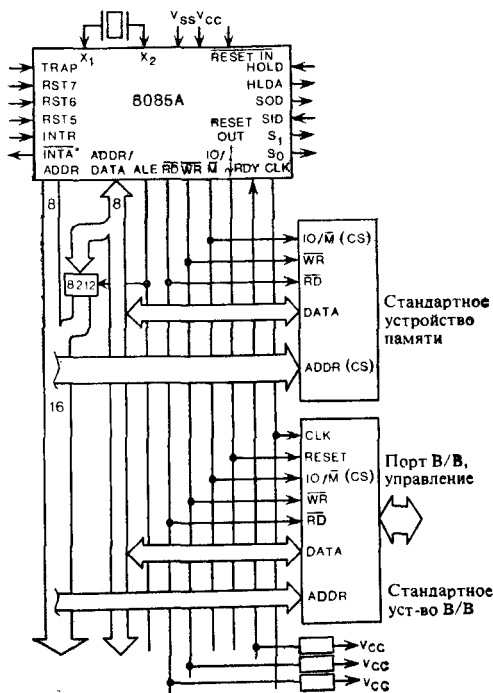


Рис. П.7. Система MCS-85. (Используются стандартные устройства памяти.)

Описание выводов ЦП Z80, Z80A

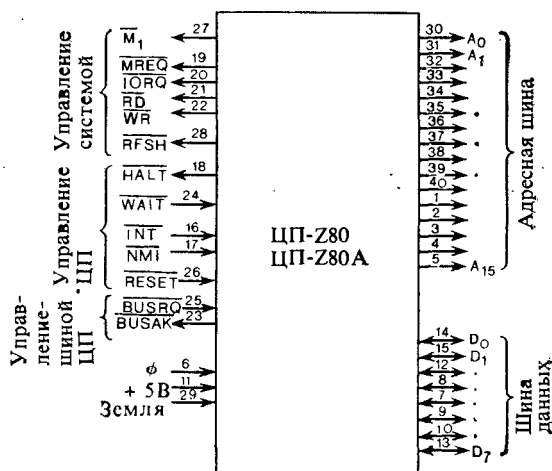


Рис. П.8. Назначение выводов микропроцессоров Z80, Z80A.

A_0 — A_{15} (адресная шина)

Выходы с тремя устойчивыми состояниями. Активный уровень сигнала — высокий. A_0 — A_{15} — выходы на 16-разрядную адресную шину. Адресная шина используется для передачи адреса памяти (может использоваться память объемом до 64 Кбайт) или адреса устройства ввода-вывода.

D_0 — D_7 (шина данных)

Входы-выходы с тремя устойчивыми состояниями. Активный уровень — высокий. К выводам D_0 — D_7 подключается восьмиразрядная двунаправленная шина данных. Шина данных используется для обмена данными между ЦП и памятью или между ЦП и устройствами ввода-вывода.

$\overline{M}_1$ (машинный цикл)

Выход. Активный сигнал — низкий. Сигнал $\overline{M}_1$ указывает, что в текущем машинном цикле проходит цикл выборки кода операции.

$\overline{MREQ}$ (запрос памяти)

Выход с тремя устойчивыми состояниями. Активный уровень сигнала — низкий. Сигнал запроса памяти указывает, что на адресной шине установлен адрес для выполнения операции записи или чтения памяти.

$\overline{IORQ}$ (запрос ввода/вывода)

Выход с тремя устойчивыми состояниями. Активный уровень сигнала — низкий. Сигнал $\overline{IORQ}$ указывает, что младший байт шины адреса содержит адрес устройства ввода-вывода, который должен быть использован при выполнении операции ввода-вывода. Кроме того, сигнал $\overline{IORQ}$ генерируется после выдачи подтверждения прерывания. Тем самым указывается, что вектор прерывания может быть помещен на шину данных.

$\overline{RD}$ (чтение из памяти)

Выход с тремя устойчивыми состояниями. Активный уровень сигнала — низкий. Сигнал $\overline{RD}$ указывает, что ЦП готов к чтению данных из памяти или

из устройства ввода. Адресованное устройство ввода или память должны использовать этот сигнал для стробирования при подаче данных на шину данных ЦП.

WR (запись в память)

Выход с тремя устойчивыми состояниями. Активный уровень сигнала — низкий. Сигнал WR указывает, что на шине данных ЦП содержатся данные, предназначенные для записи в память или для вывода на устройство вывода.

RFSH (восстановление)

Выход. Активный уровень — низкий. Сигнал RFSH указывает, что младшие 7 разрядов шины адреса содержат адрес восстановления для динамической памяти и текущий сигнал MREQ должен использоваться для выполнения восстановления динамической памяти.

HALT (останов)

Выход. Активный уровень — низкий. Сигнал HALT указывает, что ЦП Z80 выполнил команду HALT и ожидает появления либо немаскируемого, либо маскируемого прерывания, после наступления которого он продолжит функционирование. Перед выполнением «останова» ЦП пересылает в память информацию, которая потребуется для возобновления его работы.

WAIT (ожидание)

Вход. Активный уровень — низкий. Сигнал WAIT указывает ЦП Z80, что адресуемая память или адресуемое устройство ввода-вывода не готовы для выполнения передачи данных. ЦП будет находиться в состоянии ожидания до тех пор, пока этот сигнал активен.

INT (запрос на прерывание)

Вход. Активный уровень — низкий. Сигнал запроса на прерывание генерируется устройствами ввода-вывода. Сигнал запроса на прерывание будет воспринят в конце выполнения текущей команды, если триггер разрешения прерываний (IFF), управляемый внутренними программными средствами, установлен в определенное состояние.

NMI (немаскируемое прерывание)

Вход. Активный уровень — низкий. Линия запроса на немаскируемое прерывание. Это прерывание имеет более высокий приоритет по сравнению с прерыванием INT и всегда независимо от состояния триггера разрешения прерывания распознается в конце выполнения текущей команды. Сигнал NMI автоматически переводит ЦП Z80 к выполнению программы, команда которой имеет адрес 0066₁₆.

RESET

Вход. Активный уровень — низкий. При поступлении сигнала RESET выполняются следующие действия: сброс триггера разрешения прерывания, очистка счетчика команд и регистров I и R. Во время установки начального состояния адресная шина и шина данных переводятся в состояние высокого сопротивления, а для всех управляющих выходных сигналов устанавливается неактивный уровень.

BUSRQ (запрос шин)

Вход. Активный уровень — низкий. Сигнал BUSRQ имеет более высокий приоритет, чем сигнал NMI, и всегда распознается в конце текущего машинного цикла. Он используется для перевода в состояние высокого сопротивления адресной шины, шины данных и тристабильных выходов сигналов управ-

ления, после чего этими шинами могут управлять другие внешние устройства. BUSAK (подтверждение перевода шин в состояние высокого сопротивления)

Выход. Активный уровень — низкий. Сигнал BUSAK подается на запрашивающее внешнее устройство для подтверждения того, что адресная шина, шина данных и тристабильные линии шины данных перешли в состояние высокого сопротивления. Теперь внешнее устройство может управлять этими шинами.

Выборка команды

Содержимое счетчика команд подается на адресную шину непосредственно в начале машинного цикла. Через половину периода следования тактовых импульсов подается сигнал MREQ. Задний фронт сигнала MREQ может быть использован для отпирания кристалла динамической памяти. Активное состояние сигнала RD показывает, что данные из памяти уже поступили на шину данных. ЦП выполняет выборку данных, когда проходит передний фронт синхронизирующего импульса, соответствующего состоянию T_3 . Во время состояний T_3 и T_4 цикла выборки производится регенерация динамической памяти, в то же самое время ЦП дешифрирует и выполняет команду. Сигнал управления восстановлением RFSH указывает, что восстановление возможности чтения динамической памяти завершено.

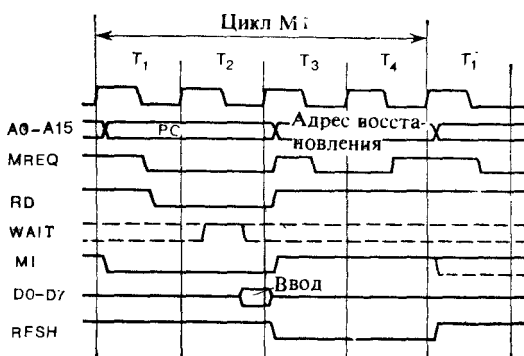


Рис. П.9. Временная диаграмма цикла M_1 .

Цикл чтения из памяти и цикл записи в память

Данная временная диаграмма циклов записи в память и чтения из памяти отличается от временной диаграммы цикла M_1 , рассмотренной выше. Однако здесь, как и в цикле выборки, используются сигналы MREQ и RD. В цикле записи сигнал MREQ активизируется, когда уровни сигналов на адресной шине стабилизировались. Поэтому он может быть непосредственно использован в качестве сигнала отпирания кристалла динамической памяти. Линия подачи сигнала WR активизируется, когда данные на шине данных уже стабилизировались. Таким образом, этот сигнал можно непосредственно использовать в качестве импульса чтения-записи для любых типов полупроводниковой памяти.

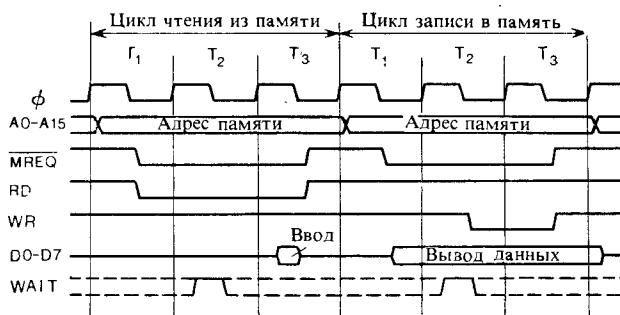


Рис. П.10. Временная диаграмма цикла чтения из памяти и цикла записи в память.

Циклы операций ввода-вывода

Рассмотрим временные диаграммы, соответствующие выполнению операции чтения данных из устройства ввода и операции записи данных в устройство вывода. Отметим, что во время выполнения операций ввода-вывода автоматически генерируется состояние ожидания, продолжающееся в течение времени T_w . Благодаря состоянию ожидания обеспечивается время, достаточное для дешифрирования адреса в порте ввода-вывода и для активизирования линии сигнала WAIT.

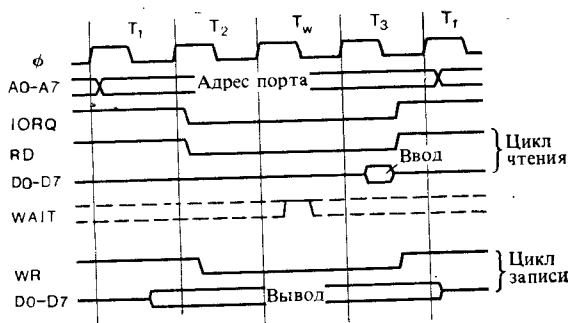


Рис. П.11. Временная диаграмма цикла операции ввода и операции вывода.

Цикл запроса на прерывание и подтверждения прерывания

ЦП проверяет наличие запроса на прерывание при прохождении переднего фронта последнего тактового импульса в цикле любой команды. Если обнаруживается запрос на прерывание, то генерируется особый цикл M_1 . В течение этого цикла M_1 активизируется сигнал IORQ (вместо сигнала MREQ), который указывает устройству, запросившему прерывание, что оно может подать на шину данных 8-разрядный вектор прерывания. В этот цикл автоматически включается два интервала ожидания, продолжительность каждого

из которых равна T^*w . Это позволяет сравнительно легко реализовать схему обработки приоритетных прерываний, подобную используемой в периферийных контроллерах микропроцессорной системы Z80.

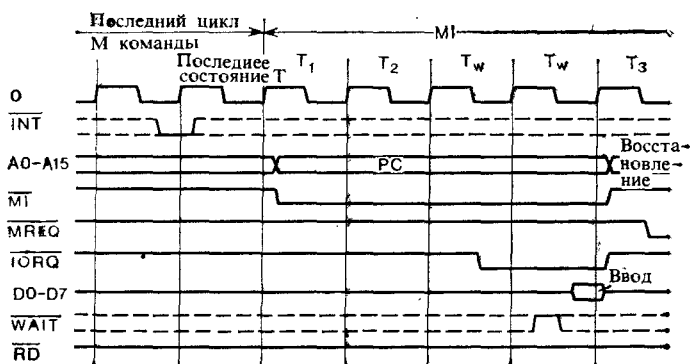


Рис. П.12. Временная диаграмма цикла запроса-подтверждения прерывания.

Допустимые предельные значения (Z80/Z80A)

Температура окружающей среды	Определяется областью применения
Температура устройства памяти	$-65 - +150^{\circ}\text{C}$
Напряжения на всех выводах по отношению к корпусу	$-0,3 - +7\text{ В}$
Мощность рассеяния	1,5 Вт

Таблица 5

Характеристики ЦП Z80A по постоянному току
 $T_A = 0-70^\circ\text{C}$, $V_{CC} = +5\text{ В} \pm 5\%$, если не оговорено особо

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{ILC}	Низкий уровень тактового напряжения	-0,3		0,45	В	
V_{INC}	Высокий уровень тактового напряжения	$V_{CC} - 0,6$		$V_{CC} + 0,3$	В	
V_{IL}	Низкий уровень входного напряжения	-0,3		0,8	В	
V_{IH}	Высокий уровень входного напряжения	2,0		V_{CC}	В	
V_{OL}	Низкий уровень выходного напряжения			0,4	В	$I_{OL} = 1,8\text{ мА}$
V_{OH}	Высокий уровень выходного напряжения	2,4			В	$I_{OH} = -250\text{ мкА}$
I_{CC}	Ток от источника			150	мА	
I_{LI}	Ток ут. на входе			10	мкА	$V_{IN} = 0 \div V_{CC}$
I_{LOH}	Ток ут. на входе с тремя состояниями			10	мкА	$V_{OUT} = 2,4 \div V_{CC}$
I_{LOL}	Ток ут. на выходе с тремя состояниями			-10	мкА	$V_{OUT} = 0,4\text{ В}$
I_{LD}	Ток ут. на шине данных в режиме ввода			± 10	мкА	$0 \leq V_{IN} \leq V_{CC}$

Характеристики ЦП Z80A по постоянному току
 $T_A = 0-70^\circ\text{C}$, $V_{CC} = +5\text{ В} \pm 5\%$, если не оговорено особо

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{ILC}	Низкий уровень тактового напряжения	-0,3		0,45	В	
V_{INC}	Высокий уровень тактового напряжения	$V_{CC} - 0,6$		$V_{CC} + 0,3$	В	
V_{IL}	Низкий уровень входного напряжения	-0,3		0,8	В	

Продолжение табл. 5

Обозначение	Параметр	Значение			Единица измерения	Условия испытаний
		минимальное	среднее	максимальное		
V_{IN}	Высокий уровень входного напряжения	2,0		V_{CC}	В	
V_{OL}	Низкий уровень выходного напряжения			0,4	В	$I_{OL}=1,8 \text{ мА}$
V_{OH}	Высокий уровень выходного напряжения	2,4			В	$I_{OH}=-250 \text{ мкА}$
I_{CC}	Ток от источника		90	200	мА	
I_{LI}	Ток ут. на входе			10	мкА	$V_{IN}=0 \div V_{CC}$
I_{LOH}	Ток ут. на входе с тремя состояниями			10	мкА	$V_{OUT}=2,4 \div V_{CC}$
I_{LOL}	Ток ут. на входе с тремя состояниями			-10	мкА	$V_{OUT}=0,4 \text{ В}$
I_{LD}	Ток ут. на шине данных в режиме ввода			± 10	мкА	$0 \leq V_{IN} \leq V_{CC}$

Таблица 6

Емкостные характеристики

 $T_A=25^\circ\text{C}$, $f=1 \text{ МГц}$.

Свободные от измерения выводы подключаются к корпусу

Обозначение	Параметр	Максимальное значение	Единица измерения
C_ϕ	Емкость на входах ϕ	35	пФ
C_{IN}	Входная емкость	5	пФ
C_{OUT}	Выходная емкость	10	пФ

ОГЛАВЛЕНИЕ

Предисловие к русскому изданию	5
Предисловие	6
Предисловие автора	7
Глава 1. Введение в архитектуру машины с 3 шинами	9
1.1. Архитектура систем с 3 шинами	9
1.2. Адресная шина системы	10
1.3. Шина данных системы	11
1.4. Шина управления системы	12
1.5. Использование архитектуры с 3 шинами	13
1.6. Запись данных в память	13
1.7. Чтение данных из памяти	16
1.8. Запись данных в устройство вывода	17
1.9. Чтение данных с устройства ввода	19
1.10. Операции с внутренними регистрами	20
1.11. Выполнение команд в системе с 3 шинами	20
1.12. Управление синхронизацией системы	21
1.13. Выводы	22
Глава 2. Построение систем с 3 шинами на базе устройств 8080, 8085 Z80 и 6800	23
2.1. Пояснения к адресной шине системы	23
2.2. Выбор требуемых буферов адреса	27
2.3. Адресная шина микропроцессора 8080	28
2.4. Использование буферов адреса	30
2.5. Адресная шина Z80	34
2.6. Адресная шина 6800	35
2.7. Адресная шина микропроцессора 8085	35
2.8. Пояснения к шине данных системы	40
2.9. Буферизованная шина данных микропроцессора 8080 а.	42
2.10. Буферизованная шина данных микропроцессора Z80	46
2.11. Буферизованная шина данных микропроцессора 6800	46
2.12. Буферизование шины данных микропроцессора 8085	48
2.13. Пояснения к шине управления системы	48
2.14. Шина управления системы на базе микропроцессора 8080	49
2.15. Фиксатор состояния микропроцессора 8080	50
2.16. Шина управления системы на базе Z80	52
2.17. Шина управления системы на базе 8085	52
2.18. Шина управления системы на базе 6800	54
2.19. Распределение памяти	54
2.20. Выводы	61
Глава 3. Генераторы тактовых импульсов и интерфейс памяти для микропроцессорных систем с 3 шинами	62
3.1. Генератор тактовых импульсов для микропроцессора 8080	62
3.2. Генератор тактовых импульсов для микропроцессора 8085	66

3.3. Генератор тактовых импульсов для микропроцессора Z80 . . .	67
3.4. Генератор тактовых импульсов для микропроцессора 6800 . . .	67
3.5. Выводы . . .	71
3.6. Интерфейс памяти в микропроцессорных системах с 3 шинами . . .	71
3.7. ПЗУ системы . . .	72
3.8. Интерфейс ОЗУ . . .	77
3.9. Чтение данных из ОЗУ . . .	81
3.10. Запись данных в ОЗУ . . .	82
3.11. Интерфейс ОЗУ с общим входом и выходом . . .	85
3.12. Микропроцессор как системный контроллер . . .	87
3.13. Подготовка микропроцессора 8080 для работы в режиме системного контроллера . . .	88
3.14. Установка начального состояния микропроцессора 8080 . . .	88
3.15. Подготовка микропроцессора 8085 для работы в режиме системного контроллера . . .	91
3.16. Подготовка микропроцессора Z80 для работы в режиме системного контроллера . . .	91
3.17. Начальная установка микропроцессора Z80 . . .	91
3.18. Подготовка микропроцессора 6800 для работы в режиме системного контроллера . . .	94
3.19. Начальная установка микропроцессора 6800 . . .	94
3.20. Выводы . . .	96
Глава 4. Интерфейс устройств ввода-вывода в микропроцессорных системах с 3 шинами . . .	98
4.1. Программное обеспечение клавишного пульта . . .	99
4.2. Аппаратные средства, необходимые для реализации клавишного пульта . . .	100
4.3. Сигналы горизонтальных линий . . .	104
4.4. Распознавание сигналов на выходах клавишного пульта . . .	107
4.5. Выводы . . .	108
4.6. Цифровой индикатор . . .	109
4.7. Программное управление клавишным пультом . . .	111
4.8. Программный метод формирования сигналов на входах матрицы клавишного пульта . . .	111
4.9. Опрос выходных линий клавишного пульта с помощью программных средств . . .	117
4.10. Вычисление весового значения ключа . . .	120
4.11. Программные средства учета эффекта вибрации клавиатуры . . .	123
4.12. Инициализация программы . . .	130
4.13. Выводы . . .	144
Глава 5. Применение метода тестирования статическими сигналами для отладки аппаратных средств микропроцессорных систем . . .	145
5.1. Идея метода тестирования статическими сигналами . . .	145
5.2. Аппаратные средства устройства тестирования статическими сигналами . . .	147
5.3. Адресные линии . . .	147
5.4. Сигналы управления . . .	149
5.5. Линии шины данных . . .	150
5.6. Применение устройства тестирования статическими сигналами . . .	153
5.7. Выбор точки начала контроля . . .	153
5.8. Проверка адресной шины . . .	158
5.9. Проверка шины управления . . .	161
5.10. Проверка правильности подачи сигналов выбора кристаллов и сигналов разрешения записи в память . . .	163
5.11. Запись и чтение данных из устройств ввода-вывода . . .	165

5.12. Проверка функционирования схемы клавишного пульта с помощью устройства тестирования	167
5.13. Выводы	168
Глава 6. Прерывания, режим ожидания и режим прямого доступа к памяти в микропроцессорах 8080, 8085, 6800 и Z80	170
6.1. Основные представления о прерываниях	170
6.2. Прерывания в микропроцессоре 8080	172
6.3. Прерывания в микропроцессоре 8085	183
6.4. Прерывания в микропроцессоре Z80	187
6.5. Прерывания в микропроцессоре 6800	189
6.6. Способы реализации режима «ОЖИДАНИЕ»	190
6.7. Перевод в состояние ожидания микропроцессоров 8080, 8085 и Z80	191
6.8. Перевод в состояние ожидания микропроцессора 6800	196
6.9. Прямой доступ к памяти в микропроцессорах 8080, 8085 Z80 и 6800	199
6.10. Прямой доступ к памяти в микропроцессоре 8080	200
6.11. Прямой доступ к памяти в микропроцессоре 8085	202
6.12. Прямой доступ к памяти в микропроцессоре Z80	202
6.13. Прямой доступ к памяти в микропроцессоре 6800	203
6.14. Выводы	204
Глава 7. Программирование перепрограммируемых постоянных запоминающих устройств	205
7.1. Общие представления о перепрограммируемых постоянных запоминающих устройствах	205
7.2. Стирание информации в перепрограммируемых постоянных запоминающих устройствах	206
7.3. Программирование ППЗУ 2708	207
7.4. Импульс программирования ППЗУ 2708	211
7.5. Выводы	214
Глава 8. Технические средства устройства программирования ППЗУ	215
8.1. Общее описание системы	215
8.2. Специфические функции системы	216
8.3. Вспомогательные технические средства	217
8.4. Клавишное устройство ввода-вывода	218
8.5. Технические средства устройства отображения	220
8.6. Постоянная память системы	220
8.7. Оперативная память системы	220
8.8. Интерфейс программирования 2708	222
8.9. Технические средства ввода и вывода данных	223
8.10. Средства формирования импульса программирования	225
8.11. Средства управления уровнем напряжения на выводе $\overline{CS}/WE$	227
8.12. Источник питания для ППЗУ 2708	227
8.13. Соединение панели программирования с системой	228
8.14. Использование микропроцессора 6800 в качестве управляющего устройства	230
8.15. Выводы по техническим средствам	230
Глава 9. Проектирование программного обеспечения для управления микропроцессорной системой	236
9.1. Начальный этап	236
9.2. Общее представление о программном обеспечении	240
9.3. Программы ввода информации с клавишного пульта	241
9.4. Главная управляющая программа	249
9.5. Программные средства реализации функции установки адреса	254

9.6. Программные средства реализации функции СТИРАНИЯ ОЗУ	260
9.7. Программные средства реализации функции программирования	260
9.8. Программные средства реализации функции верификации	266
9.9. Программные средства реализации функции копирования	282
9.10. Выводы по программным средствам	282
Глава 10. Отладка технических средств системы	283
10.1. Тестирование посредством статических сигналов	283
10.2. Устройство тестирования посредством статических сигналов для микропроцессора 8085	285
10.3. Замечания по отладке технических средств системы	291
10.4. Отладка адресной шины системы	292
10.5. Отладка шины управления системы	297
10.6. Отладка шины данных системы	299
10.7. Проверка схем дешифрирования выбора памяти	301
10.8. Проверка интерфейса ПЗУ системы	302
10.9. Проверка интерфейса ОЗУ системы	306
10.10. Отладка интерфейса клавишного пульта	311
10.11. Отладка интерфейса устройства отображения	311
10.12. Отладка схем подключения ППЗУ к системе	311
10.13. Проверка схем формирования импульса программирования	318
10.14. Проверка функционирования схем ВЫБОРКА БЛОКА и РАЗРЕ- ШЕНИЯ ЗАПИСИ	320
10.15. Выводы	323
Приложение	324
Назначение выводов корпуса микропроцессора INTEL 8080A	324
Допустимые предельные значения 8080A	327
Функциональное описание 8085A/8085A-2	330
Прерывания и последовательный ввод-вывод	330
Допустимые предельные значения (8085A/8085A-2)	332
Описание выводов ЦП Z80, Z80A	334
Выборка команды	336
Цикл чтения из памяти и цикл записи в память	336
Циклы операций ввода-вывода	337
Цикл запроса на прерывание и подтверждения прерывания	337
Допустимые предельные значения (Z80/Z80A)	338

УВАЖАЕМЫЙ ЧИТАТЕЛЬ!

Ваши замечания о содержании книги, ее оформлении, качестве перевода и другие просим присылать по адресу: 129820, Москва, И-110, ГСП, 1-й Рижский пер., 2, издательство «Мир».

